МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

О. В. Стусь

МАТЕМАТИЧНА ЛОГІКА ТА ТЕОРІЯ АЛГОРИТМІВ: ЛЕКЦІЇ

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для студентів,
які навчаються за спеціальністю 124 «Системний аналіз»

Київ
КПІ ім. Ігоря Сікорського
2017

Рецензент: *Ільєнко А. Б.,* канд. фіз.-мат. наук, доц.

Відповідальний редактор *Богданський Ю. В.,* д-р фіз.-мат. наук, проф.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 2 від 19.10.2017 р.) за поданням Вченої ради ІПСА (протокол № 8 від 25.09.2017 р.)

Електронне мережне навчальне видання

Стусь Олександр Вікторович, канд. фіз.-мат. наук.

МАТЕМАТИЧНА ЛОГІКА ТА ТЕОРІЯ АЛГОРИТМІВ:
ЛЕКЦІЇ

Математична логіка та теорія алгоритмів: Лекції [Електронний ресурс] : навч. посіб. для студ. спеціальності 124 «Системний аналіз» / О. В. Стусь ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 0,8 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2017. – 150 с.

Посібник містить теоретичні відомості із традиційних розділів дисципліни «Математична логіка та теорія алгоритмів» – алгоритмічні машини, рекурсивні функції, формальні логічні теорії, алгебра предикатів, автоматичне доведення теорем, формальна арифметика.

Для студентів математичних і технічних спеціальностей університетів. яким знайомі базові поняття дискретної математики та математичного аналізу.

Зміст

Вступ

«Математична логіка та теорія алгоритмів» – одна з основних фундаментальних дисциплін у загальнонауковій підготовці студентів за спеціальністю 124 «Системний аналіз». Дисципліна «Математична логіка і теорія алгоритмів» є базовою для таких дисциплін, як «Мови програмування» (розділ «Теорія алгоритмів»), «Спеціалізовані мови програмування» (розділ «Алгебра предикатів»), «Експертні системи» (розділ «Автоматичне доведення теорем) та інших. При вивченні курсу використовуються основні результати дисциплін «Дискретна математика», «Математичний аналіз» та «Лінійна алгебра».

У навчальному посібнику подано теоретичний матеріал за розділами: «Алгоритмічні машини», «Рекурсивні функції», «Формальні логічні теорії», «Алгебра предикатів», «Автоматичне доведення теорем», «Формальна арифметика». Матеріал, згідно робочої програми дисципліни «Математична логіка і теорія алгоритмів» Інституту прикладного системного аналізу, розраховано на викладання протягом п’ятнадцяти лекцій у другому та третьому семестрі.

Розділи: «Алгоритмічні машини», «Алгебра предикатів», «Автоматичне доведення теорем» згідно робочої програми входять до дисципліни «Дискретна математика» для підготовки студентів за спеціальністю 122 «Комп’ютерні науки». Матеріал, згідно робочої програми дисципліни «Дискретна математика» Інституту прикладного системного аналізу, розраховано на викладання протягом дев’яти лекцій у другому семестрі.

Означення та теореми проілюстровано прикладами. Прості твердження та твердження, що можуть бути доведеними за аналогією, запропоновані як вправи для самостійної роботи.

Послідовність і стиль подання матеріалу повністю відповідають робочій програмі та задовольняють потреби суміжних і прикладних дисциплін.

Розділ 1. Алгоритмічні машини

1.1. Поняття алгоритму

Означення 1.1. (неформальне означення алгоритму). Алгоритм – покрокова процедура, яка дозволяє розв'язувати будь-яку задачу з даного класу задач.

Властивості алгоритму:

1. Дискретність – дії виконуються покроково;
2. Скінченність – алгоритм завершує роботу за скінченну кількість кроків;
3. Виконуваність – операцію можна виконати на кожному конкретному кроці;
4. Детермінованість – результат алгоритму має однозначно визначатися вхідними даними;
5. Масовість – алгоритм повинен розв'язувати всі задачі з даного класу.

Одним із завдань курсу є математичне обґрунтування поняття алгоритму та вирішення суміжних проблем. В першу чергу виникає питання, для яких задач можна побудувати алгоритм та дати формальне визначення алгоритму.

Далі ми будемо розглядати кілька еквівалентних між собою абстрактних алгоритмів (алгоритмічних машин). Ці машини можна вважати формальним визначенням алгоритму. І головне значення цих алгоритмів – які задачі можна розв'язувати за допомогою них (поняття ефективної обчислюваності). Тому вигідно кожну задачу розбивати на

певну кількість як можна простіших кроків. У цьому і полягає ідея побудови кожної з нижченаведених алгоритмічних машин.

1.2. Машина Тьюрінга

1.2.1. Опис машини Тьюрінга

Машина Тьюрінга[1] – абстрактний алгоритм, який представляється у вигляді нескінченної в обидва боки стрічки, розділеної на комірки (рис. 1.1). У кожній комірці може міститися не більше одного символу з деякої скінченної множини (зовнішній алфавіт). Якщо комірка порожня, то вважають, що там міститься порожній символ, який позначається Λ (лямбда). Крім того, є курсор (вказівник, голівка, яка зчитує), який у кожний момент часу вказує на одну з комірок, яку будемо називати поточною. Машина в кожний момент часу зчитує вміст поточної комірки. Вміст решти комірок недоступний. Робота машини Тьюрінга відбувається покроково. У кожен момент часу машина знаходиться в одному зі своїх (внутрішніх) станів, серед яких будемо виділяти початковий (частіше всього позначається через «q_0») та заключний (прийнято позначати через «!»).

[1] Алан Матісон Тюрінг (1912 – 1954) – англійський математик, логік, криптограф. Запропонована ним у 1936 році абстрактна обчислювальна «машина Тюрінга» дозволила формалізувати поняття алгоритму і до теперішнього часу використовується в багатьох теоретичних та практичних дослідженнях.

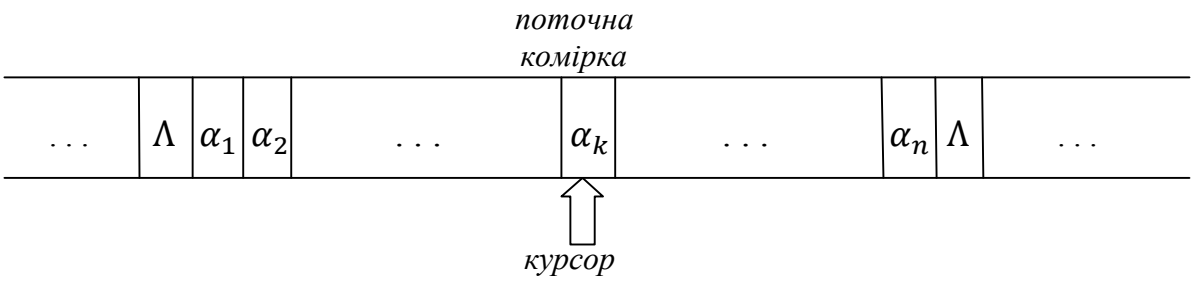


Рис. 1.1. Загальний вигляд машини Тьюрінга

Тут $\alpha_1, \alpha_2, \cdots \alpha_k, \cdots, \alpha_n$ – символи зовнішнього алфавіту.

Машина Тьюрінга задається:

1. Зовнішній алфавіт – непорожня скінченна множина символів, які можуть бути записані в комірки: $A \neq \emptyset,\ |A| < \infty$.
2. Внутрішній алфавіт – непорожня скінченна множина внутрішніх станів машини: $Q \neq \emptyset,\ |Q| < \infty$. В множині Q виокремлюється початковий стан (частіше за все позначається через q_0) та заключний стан (частіше за все позначається через !).
3. Множина команд – непорожня скінченна множина. Кожна команда визначає дію машини при певному поточному стані та поточному символі: $P \neq \emptyset,\ |P| < \infty$.

   Кожна команда має наступний вигляд:

$$q_m\alpha \rightarrow q_n\beta \begin{bmatrix} R \\ L \\ S \end{bmatrix},$$

де q_m – поточний стан, α – поточний символ.

Зауваження 1.1. Забороняється поява різних команд з однаковою лівою частиною.

Дія команди на даному кроці:

Нехай на даному кроці машина знаходиться в стані q_m та зчитується символ α. У цьому випадку вона здійснює такі дії:

а) записує в комірку літеру β замість α;

б) переходить у новий стан q_n;

в) курсор зміщується на одну комірку вправо (R), вліво (L) або залишається на місці (S).

1.2.2. Робота машини Тьюрінга

У початковий момент часу на стрічці записана скінченна послідовність символів (вхідне слово), машина знаходиться в початковому стані (q_0) та, якщо не визначено інше, курсор зчитує перший символ вхідного слова. Далі, відповідно до списку команд, курсор рухається по стрічці і слово перетворюється. Машина Тьюрінга закінчує роботу у двох випадках:

а) здійснено перехід у заключний стан (нормальне завершення роботи);

б) відсутня команда з відповідною лівою частиною (ненормальне або аварійне закінчення роботи).

Зауваження 1.2. Наприкінці роботи машини правилом гарного тону вважається повернути курсор на початок слова.

Приклад 1.1. До слова, яке складається із скінченної кількості паличок, дописати у кінці зірочку.

1) $q_0|\rightarrow q_0|\,R$
2) $q_0\Lambda\ \rightarrow q_1 * L$
3) $q_1|\rightarrow q_1|L$
4) $q_1\Lambda \rightarrow !\,\Lambda\,R$

Пояснення. У стані q_0 ми проходимо все слово і в кінці (як тільки зустрінемо Λ) ставимо зірочку та переходимо в стан q_1. У стані q_1 ми проходимо слово справа наліво і зупиняємося на першому символі (крайня ліва паличка) і переходимо в заключний стан !.

Розберемо цей приклад покроково (табл. 1.1).:

Табл. 1.1. Покроковий протокол роботи машини Тьюрінга для прикладу 1.1.

№ кроку	Команда	Стан	Стрічка
0	–	q_0	$\Lambda\underline{\vert}\vert\vert\Lambda$
1	1)	q_0	$\Lambda\vert\underline{\vert}\vert\Lambda$
2	1)	q_0	$\Lambda\vert\vert\underline{\vert}\Lambda$
3	1)	q_0	$\Lambda\vert\vert\vert\underline{\Lambda}$
4	2)	q_1	$\Lambda\vert\vert\underline{\vert} * \Lambda$
5	3)	q_1	$\Lambda\vert\underline{\vert}\vert * \Lambda$
6	3)	q_1	$\Lambda\underline{\vert}\vert\vert * \Lambda$
7	3)	q_1	$\underline{\Lambda}\vert\vert\vert * \Lambda$
8	4)	!	$\Lambda\underline{\vert}\vert\vert * \Lambda$

Підкреслений символ на стрічці означає положення курсору.

Приклад 1.2. Обчислити $x + 1$ у двійковій системі числення.

1) $q_0\xi \to q_0\xi\ R$
2) $q_0\Lambda \to q_1\Lambda\ L$
3) $q_1 0 \to q_2 1\ L$
4) $q_1 1 \to q_1 0\ L$
5) $q_1\Lambda \to !\,1\ S$
6) $q_2\xi \to\ q_2\xi\ L$
7) $q_2\Lambda \to !\,\Lambda\ R$

$\xi\ \in \{0; 1\}$ використовується для скорочення запису. Тобто команда $q_0\xi \to q_0\xi\ R$ означає дві команди: $q_0 0 \to q_0 0\ R$ та $q_0 1 \to q_0 1\ R$.

Пояснення. У стані q_0 ми проходимо число до кінця, переходимо в стан q_1 і стаємо на останню цифру. Якщо ця цифра – «1», то замість неї пишемо «0» і стан не змінюємо, бо з усіма наступними одиницями виконуємо цю саму дію. Але якщо зустрічаємо «0», то замість нього пишемо «1» і змінюємо стан на q_2. У стані q_2 ми проходимо число справа наліво і зупиняємося на першій цифрі отриманого числа. Окремий випадок, коли ми зустрінемо Λ у стані q_1. Цей випадок означає, що ми маємо справу з числом вигляду $11 \ldots 1$. Тут всі «1» замінилися на «0», і тоді залишається замість Λ поставити «1» і перейти в заключний стан.

1.2.3. Способи задання машини Тьюрінга

Машина Тьюрінга може задаватися:

1. списком команд (див. вище);
2. табличним способом;

3. графічним способом.

Табличний спосіб задання машини Тьюрінга має вигляд (табл. 1.2):

Табл. 1.2. Табличний спосіб задання машини Тьюрінга

	q_0	q_1	…	q_n	…	q_N	!
α_1			…		…		
α_2			…		…		
…	…	…	…	…	…	…	
α_i			…	$q_m\alpha_j\ R$	…		
…	…	…	…	…	…	…	
α_M			…		…		
Λ			…		…		

тут $Q = \{q_0, q_1, \dots, q_N, !\}$,
$A = \{\alpha_1, \alpha_2, \dots, \alpha_M\}$.

Команда $q_n\alpha_i \to q_m\alpha_j\ R$ відображена в табл. 1.2. У відповідну клітинку записується права частина команди.

Графічний спосіб: Машина Тьюрінга зображується у вигляді орієнтованого міченого мультторграфа, де множина вершин співпадає з множиною станів, а кожна команда зображується у вигляді ребра. Наприклад, команда $q_n\alpha_i \to q_m\alpha_j\ R$ зображується таким чином (рис. 1.2):

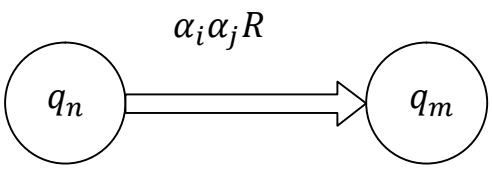


Рис. 1.2. Зоброження команди машини Тьюрінга

Продемонструємо табличний та графічний спосіб для розв'язання задачі з прикладу 1.2 (табл. 1.3 та рис. 1.3):

Табл. 1.3. Табличний спосіб задання машини Тьюрінга для прикладу 1.2.

	q_0	q_1	q_2	!
0	$q_0 0R$	$q_2 1L$	$q_2 0L$	
1	$q_0 1R$	$q_1 0L$	$q_2 1L$	
Λ	$q_1 \Lambda L$	$1!\,S$	$\Lambda!\,R$	

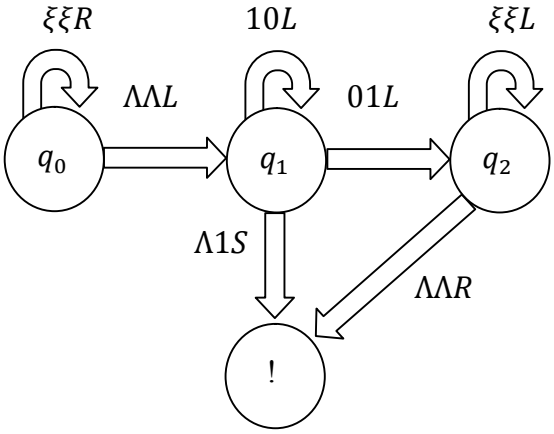


Рис. 1.3. Графічний спосіб задання машини Тьюрінга для прикладу 1.2

Приклад 1.3. Скопіювати слово в алфавіті $\{a, b\}$. Вхідне слово: w. Вихідне слово: $w = w$.

Побудуємо граф машини Тьюрінга для розв'язання даної задачі (рис 1.4). Позначимо $\xi \epsilon \{a, b\}$, $\tilde{\xi} \epsilon \{\tilde{a}, \tilde{b}\}$.

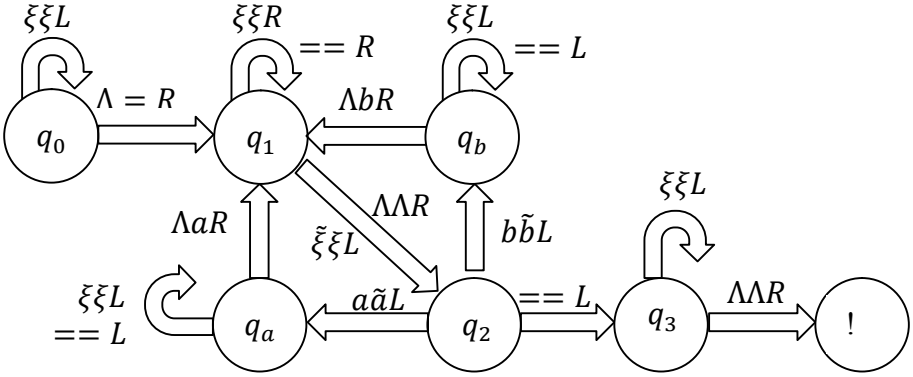


Рис. 1.4. Граф для прикладу 1.3.

Пояснення. Слово будемо копіювати справа наліво. У стані q_0 ставимо знак «=» перед словом і більше цей стан використовувати не будемо. Переходимо в стан q_1. У цьому стані ми проходимо слова зліва

направо до крайнього не скопійованого символу. Першого разу це буде останній символ слова, а наступні рази – символ, який стоїть лівіше від відміченого ($\tilde{\xi}$). Знімаємо мітку і переходимо в стан q_2. У стані q_2 запам'ятовуємо символ, який будемо копіювати, переходячи в стан q_a чи q_b, а також відмічаємо цей символ (ставимо хвильку). У випадку, коли всі символи вже скопійовані (у стані q_2 потрапили на символ «=»), переходимо в стан q_3 і рухаємося вліво на початок результуючого слова. У стані q_a (q_b) проходимо слово вліво та на вільне місце записуємо літеру a (b), повертаємося в стан q_1. Продовжуємо процедуру далі, поки не скопіюємо все слово w. Алгоритм буде також працювати у випадку порожнього слова. Вихідним словом буде «=».

Вправа 1.1. Побудувати машину Тьюрінга, яка:

1. Слово в алфавіті $\{a, b\}$ записує в дужках. Вхідне слово: w. Вихідне слово: (w);
2. Обчислює $x - 1$ у двійковій системі $(x > 0)$;
3. Переводить число з унарної у двійкову систему;
4. Переводить число з двійкової в унарну систему;
5. Перевіряє, чи є слово в алфавіті $\{a, b\}$ паліндромом (читається однаково зліва направо та справа наліво).

1.2.4. Композиція машин Тьюрінга

Для розв'язання складних задач за допомогою машини Тьюрінга доцільно буває будувати машини для окремих підзадач, а потім їх зв'язувати в одну машину.

Означення 1.2. Нехай T_1 та T_2 – дві машини Тьюрінга, у яких множини внутрішніх станів не перетинаються ($Q_1 \cap Q_2 = \emptyset$). Композицією машин T_1 та T_2 називається машина $T = T_2 \circ T_1$, робота якої полягає в тому, що спочатку працює T_1, а потім T_2 (вихідне слово T_1 є вхідним для T_2). $T(w) = T_2\big(T_1(w)\big) = (T_2 \circ T_1)(w)$.

Для реалізації композиції машин Тьюрінга T_1 та T_2 достатньо ототожнити кінцевий стан T_1 та початковий T_2 (рис 1.5 а))

Композицію машин Тьюрінга можна узагальнити для скінченної кількості машин: $T = T_n \circ T_{n-1} \circ \ldots \circ T_2 \circ T_1$.

У більш складних випадках (див. рис. 1.5 б) і в)) необхідно визначити для машини T_1 два (можливо і більше) заключних стани, у кожен з яких переходимо у випадку виконання певної умови.

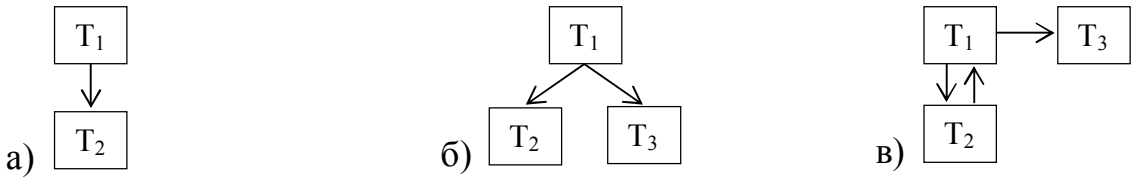


Рис. 1.5. Композиція машин Тьюрінга.

1.2.5. Різновиди машин Тьюрінга

Модель машин Тьюрінга допускає різноманітні узагальнення. Розглядаються машини Тьюрінга з довільною кількістю стрічок та багатовимірними стрічками з різними обмеженнями. Також розглядається напівнескінченна стрічка (необмежена в один бік). Недетермінована

машина Тьюрінга може перебувати в кількох станах одночасно. Всі ці машини еквівалентні звичайній машині Тьюрінга.

Крім машини Тьюрінга існують інші алгоритми їй еквівалентні.

1.3. Нормальні алгоритми Маркова

1.3.1. Опис нормального алгоритму Маркова

Нормальний алгоритм Маркова[2] (НАМ) задається наступним чином.

Нехай $\mathcal{A}$ – алфавіт (непорожня скінченна множина символів) $\mathcal{A} \neq \emptyset$, $|\mathcal{A}| < \infty$. НАМ записується у вигляді лінійно впорядкованої множини підстановок P ($P \neq \emptyset$, $|P| < \infty$).

$$P = \{\alpha_i \longrightarrow \beta_i \mid \alpha_i, \beta_i \in \mathcal{A}^* \; i = \overrightarrow{1,n}\}$$

Тут $\mathcal{A}^*$ – множина всіх слів над алфавітом $\mathcal{A}$. Слово – скінченна послідовність символів з даного алфавіту. $\mathcal{A}^*$ містить також порожнє слово (не містить жодного символу), яке будемо позначати через ε. У множині P обов'язкова наявність хоча б однієї заключної підстановки, яку будемо позначати через $\alpha_k \dot{\longrightarrow} \beta_k$.

Окремо взята підстановка $\alpha_i \longrightarrow \beta_i$ застосовується у випадку, коли поточне слово містить α_i як підслово. Застосування підстановки полягає в тому, що перше входження α_i замінюється на β_i. Порожнє підслово ε

[2] Марков Андрій Андрійович (молодший, 1903 – 1979) – радянський математик, основоположник радянської школи конструктивної математики, автор поняття нормального алгоритму.

міститься між будь-якими літерами даного слова і перше його входження міститься перед першою літерою слова.

1.3.2. Робота нормального алгоритму Маркова

На кожному кроці до поточного слова використовується завжди перша підстановка, яку можна застосувати.

Алгоритм закінчує роботу у двох випадках:

а) використали одну із заключних підстановок (нормальне завершення роботи);

б) жодну з підстановок не можна застосувати до поточного слова (ненормальне, або аварійне, завершення роботи).

Приклад 1.4. Анулювати слово в алфавіті $\{a, b\}$.

1) $a \longrightarrow \varepsilon$
2) $b \longrightarrow \varepsilon$
3) $\varepsilon \dot{\longrightarrow} \varepsilon$

Пояснення. Спочатку застосовується підстановка 1). Видаляються всі літери a. Потім підстановка 2). Видаляються всі літери b. І після цього підстановка 3) (заключна).

Розберемо цей приклад покроково для слова $aaba$ (табл. 1.4).

Табл. 1.4. Покроковий протокол роботи НАМ для прикладу 1.4.

№ кроку	Підстановка	Слово
0	–	$\underline{a}aba$
1	1)	$\underline{a}ba$
2	1)	$b\underline{a}$
3	1)	$\underline{b}$
4	2)	$\underline{\varepsilon}$
5	3)	ε

У цій та наступних таблицях для протоколів роботи НАМ підкреслений символ – перше входження підслова для виконання наступного кроку.

Приклад 1.5. Інвертувати слово в алфавіті $\{a, b\}$. Замість a пишеться b, а замість b пишеться a.

1) $* a \longrightarrow b *$
2) $* b \longrightarrow a *$
3) $*\dot{\longrightarrow} \varepsilon$
4) $\varepsilon \longrightarrow *$

Пояснення. У вхідному слові зірочки немає, тому використовується підстановка 4). Ставимо зірочку на початку слова. Потім, використовуючи підстановки 1) та 2), рухаємо зірочку вправо і замінюємо по дорозі кожну літеру a на b і літеру b на a. Якщо після зірочки немає жодної літери, застосовуємо 3). Це заключна підстановка, тому закінчуємо роботу алгоритму.

Розберемо цей приклад покроково для слова $abaa$ (табл. 1.5).

Табл. 1.5. Покроковий протокол роботи НАМ для прикладу 1.5.

№ кроку	Підстановка	Слово
0	–	$\underline{\varepsilon}abaa$
1	4)	$\underline{*\,a}baa$
2	1)	$b\,\underline{*\,b}aa$
3	2)	$ba\,\underline{*\,a}a$
4	1)	$bab\,\underline{*\,a}$
5	1)	$babb\,\underline{*}$
6	3)	$babb$

Приклад 1.6. Обертання слова в алфавіті $\{a, b\}$. Якщо вхідне слово $w = \alpha_1\alpha_2 \cdots \alpha_k$, $k \geq 0$, то оберненим до нього буде слово $w^R = \alpha_k\alpha_{k-1} \cdots \alpha_1$. Для скорочення запису введемо $\xi, \eta \in \{a, b\}$ і $\tilde{\eta} \in \{\tilde{a}, \tilde{b}\}$.

1) $\xi\tilde{\eta} \longrightarrow \tilde{\eta}\xi$
2) $\tilde{\eta} \longrightarrow \eta$
3) $*\,a \longrightarrow \tilde{a}\,*$
4) $*\,b \longrightarrow \tilde{b}\,*$
5) $*\dot{\longrightarrow} \varepsilon$
6) $\varepsilon \longrightarrow *$

Пояснення. Основний принцип даного алгоритму полягає в тому, що, рухаючи зірочку вправо по слову, ми відмічаємо літеру хвилькою, яку проштовхуємо на початок слова. Тобто якщо в початковому слові є літера з хвилькою та зірочка, то спочатку літера з хвилькою просувається вперед (підстановка 1)) і потім хвилька зникає (підстановка 2)). Якщо літери з хвилькою немає, але є зірочка, то намагаємося застосувати підстановки 3) або 4). У позитивному випадку зірочка просувається на один символ вправо і цей символ відмічається хвилькою. Далі за допомогою підстановок 1) та 2) літера з хвилькою просувається вперед і хвилька витирається.

Якщо ж підстановки 3) та 4) не можна використати, то виконуємо заключну підстановку 5), витираємо зірочку та закінчуємо роботу. Якщо немає зірочки чи літери з хвилькою, виконуємо підстановку 6).

Розберемо цей приклад покроково для слова abb (табл. 1.6).

Табл. 1.6. Покроковий протокол роботи НАМ для прикладу 1.6.

№ кроку	Підстановка	Слово
0	–	$\underline{\varepsilon}abb$
1	6)	$\underline{*\,a}bb$
2	3)	$\underline{\tilde{a}} * bb$
3	2)	$a\,\underline{*\,b}b$
4	4)	$\underline{a\tilde{b}} * b$
5	1)	$\underline{\tilde{b}}a * b$
6	2)	$ba\,\underline{*\,b}$
7	4)	$b\underline{a\tilde{b}}\,*$
8	1)	$\underline{b\tilde{b}}a\,*$
9	1)	$\underline{\tilde{b}}ba\,*$
10	2)	$bba\,\underline{*}$
11	5)	bba

Вправа 1.2. Побудувати НАМ, який:

1. Визначає останній символ слова в алфавіті $\{a, b\}$;
2. Порівнює кількість входжень літер a та b у слові в алфавіті $\{a, b\}$;
3. Переводить число з унарної у двійкову систему;
4. Переводить число з двійкової в унарну систему;
5. Обчислює $x + 1$ у двійковій системі.

1.3.3. Композиція нормальних алгоритмів Маркова

Як і для машин Тьюрінга, для алгоритмів Маркова також реалізується композиція, але трохи складніше. Тут для різних НАМ потрібно використовувати алфавіти, які не перетинаються, і перехід від одного алгоритму до іншого відбувається перейменуванням символів поточного слова.

1.3.4. Еквівалентність машини Тьюрінга та нормальних алгоритмів Маркова

Під еквівалентністю двох, або більше алгоритмів слід розуміти те, що кожна задача, яка розв'язується одним з алгоритмів, буде розв'язуватися всіма іншими.

Теорема 1.1. За класом задач, які розв'язуються, НАМ та машина Тьюрінга еквівалентні.

Доведення.

I. По заданій МТ будуємо НАМ.

Алфавіт НАМ: $A_1 = A \cup \{*_i \mid q_i \epsilon\, Q \;\; i = \overrightarrow{1,n}\}$, A – алфавіт МТ.

По кожній команді МТ будуємо підстановку для НАМ (табл. 1.7).

Табл. 1.7. Перехід від машини Тьюрінга до НАМ

Команда	Підстановка
$q_1 a \to q_2 b\ R$ $q_1 a \to q_2 b\ L$ $q_1 a \to q_2 b\ S$ $q_1 a \to !\, b \begin{bmatrix} R \\ L \\ S \end{bmatrix}$	$*_1\, a \longrightarrow b\, *_2$ $j\, *_1\, a \longrightarrow *_2\, jb$ $*_1\, a \longrightarrow *_2\, b$ $*_1\, a \dot{\longrightarrow} b$

Вправа 1.3. Продумати ситуацію на кінцях поточного слова.

II. По заданому НАМ будуємо МТ.

Для кожної підстановки НАМ будуємо МТ, яка перевіряє, чи можна її використати. У позитивному випадку підстановка використовується. Далі застосовуємо техніку композицій машин Тьюрінга за наступною схемою (рис. 1.6).

Нехай $P = \{\alpha_i \longrightarrow \beta_i \mid i = \overrightarrow{1,n}\}$, $\alpha_k \dot{\longrightarrow} \beta_k \in P$.

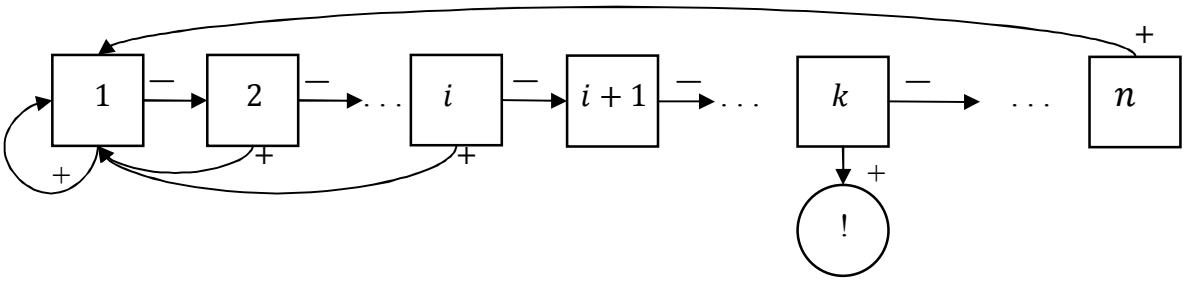


Рис.1.6. Схема побудови МТ по заданому НАМ

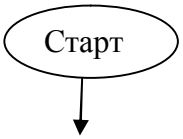
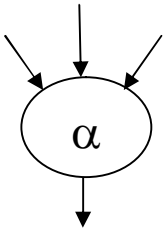
Кінець доведення.

1.4. Блок-схема Поста

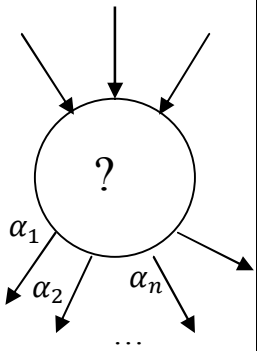
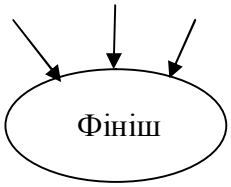
1.4.1. Опис блок-схеми Поста

Блок-схема Поста задається наступним чином. Нехай $\mathcal{A}$ – алфавіт ($\mathcal{A} \neq \emptyset$, $|\mathcal{A}| < \infty$). БСП – алгоритмічна машина, яка задається у вигляді міченого орієнтованого мультиграфа, який може містити 4 типи вершин. На кожному кроці керування знаходиться на деякій вершині (див. табл. 1.8).

Табл. 1.8. Таблиця вершин для БСП

№	Назва	Позначення	Дія	Примітка
1	Стартова		Жодна дія не виконується, означає початок роботи алгоритму.	Одна вершина для всього алгоритму, вхідних ребер немає, вихідне – одне.
2	Породжуюча (генеруюча) вершина		Записуємо символ $\alpha \in \mathcal{A}$ у кінець поточного слова.	Вхідних ребер може бути багато, але мінімум одне, вихідне ребро – одне.

Продовження табл. 1.8.

№	Назва	Позначення	Дія	Примітка
3	Тестова вершина		Видаляємо перший символ. Керування передаємо по ребру, що відповідає цьому символу.	$? \notin \mathcal{A}$, $\mathcal{A} = \{\alpha_1 \dots \alpha_n\}$. Вхідних ребер не менше одного, вихідних не більше $n+1$, де $n = \lvert A \rvert$. Ще одне ребро для випадку порожнього слова. Якщо деякий символ не зустрічається при керуванні даної вершини, то відповідне ребро можна не малювати.
4	Заключна вершина		Жодна дія не виконується, означає закінчення роботи алгоритму.	Вхідних ребер – не менше одного, вихідних – немає. Можлива наявність кількох заключних вершин.

1.4.2. Робота блок-схеми Поста

На вхід задається вхідне слово скінченної довжини із заданого алфавіту $\mathcal{A}$. БСП працює покроково. На кожному кроці рівно одна вершина є поточною. В початковий момент часу поточною є стартова вершина. При переході до наступного кроку над словом проводиться деяка дія відповідно до поточної вершини. Далі керування передається наступній вершині (відповідно вихідного ребра), наявність петель допускається. Алгоритм закінчує роботу, коли поточною є одна з заключних вершин.

Приклад 1.7. Збільшити кількість паличок у два рази (рис. 1.7).

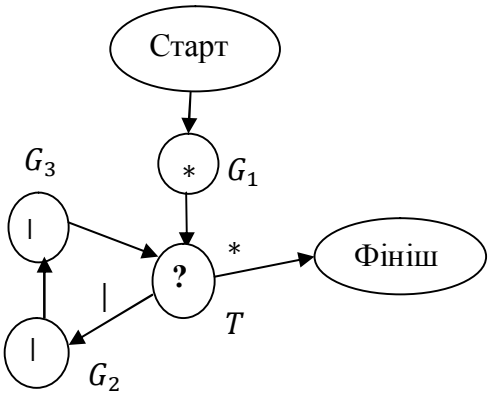


Рис.1.7. БСП для прикладу 1.7

Пояснення. Ставимо у кінці слова зірочку. А далі видаляємо першу паличку у поточному слові і в кінець записуємо дві палички. Алгоритм закінчує роботу, коли при керуванні тестової вершини зустрінеться зірочка.

.

Розберемо цей приклад покроково для слова ||| (табл. 1.9).

Табл. 1.9. Протокол роботи БСП для прикладу 1.7.

№ кроку	Поточна вершина	Поточне слово
0	*Старт*	|||
1	G_1	|||*
2	T	||*
3	G_2	||*|
4	G_3	||*||
5	T	|*||
6	G_2	|*|||
7	G_3	|*||||
8	T	*||||
9	G_2	*|||||
10	G_3	*||||||
11	T	||||||
12	*Фініш*	||||||

Приклад 1.8. З'ясувати, чи є слово в алфавіті $\{a, b\}$ поліномом $(a^n b^m, n \geq 0, m \geq 0)$. Тобто поліном є слово, у якому після літери b жодного разу не зустрічається літера a (рис1.8).

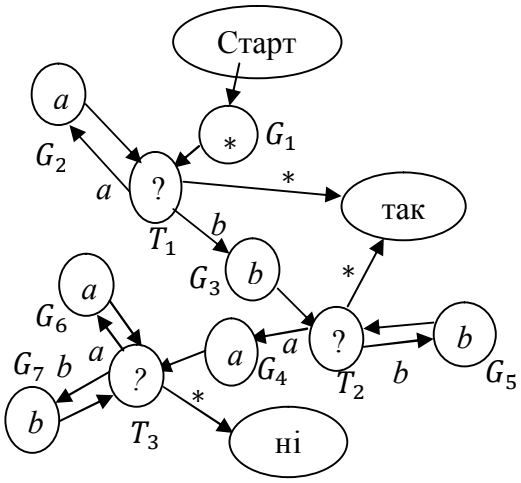


Рис.1.8. БСП для прикладу 1.8.

Пояснення. Ставимо в кінець слова зірочку (G_1) і переходимо в T_1. Промотуємо літери a (T_1 та G_2), витираємо спочатку слова та записуємо в кінець. Якщо в T_1 потрапляємо на зірочку, то вхідне слово має вигляд $a^n, n \geq 0$ і воно є поліномом. Якщо ж в T_1 потрапили на b, то дописуємо b та переходимо в T_2.

Далі промотуємо всі літери b (T_2, G_3) і у випадку потрапляння на зірочку в T_2 вхідне слово має вигляд

$a^n b^m, n \geq 0, m \geq 1$ – поліном.

Якщо ж в T_2 потрапили на літеру a, то переходимо в G_4, відновлюємо a, робимо прокрутку залишка слова (T_3, G_6, G_7) і у випадку потрапляння на зірочку закінчуємо роботу. Відповідь «ні». Після літери b є літера a.

1.4.3. Еквівалентність машини Тьюрінга, нормального алгоритму Маркова та блок-схеми Поста

Теорема 1.2. За класом задач, які розв'язуються, машина Тьюрінга, нормальні алгоритми Маркова та блок-схема Поста еквівалентні.

Схема доведення.

I. По БСП легко будується МТ. Дія кожної вершини БСП реалізується за допомогою відповідної машини Тьюрінга. А потім використовується техніка композиції.

II. Зрозуміло, що рух по слову зліва направо за допомогою БСП реалізується просто. Видаляємо символ спочатку слова та пишемо в кінець. Щоб забезпечити рух по слову справа наліво, потрібно вміти розпізнавати останній символ даного слова.

Кінець схеми доведення.

Тому розглянемо наступну задачу.

Приклад 1.9. Продублювати останній символ слова в алфавіті $\{a, b\}$. Тобто якщо вхідне слово $w = \alpha_1\alpha_2 \dots \alpha_n$, то вихідне слово буде таким: $\alpha_1\alpha_2 \dots \alpha_n * \alpha_n$ (рис. 1.9).

Пояснення. Пишемо в кінець слова зірочку (G_1) і перевіряємо перший символ слова (T_1). Якщо це зірочка, то вхідне слово було порожнім. Ставимо зірочку (G_{12}) і закінчуємо роботу.

У нетривіальних випадках переходимо в T_a, або T_b, які означатимуть, що остання прочитана літера була a, чи b відповідно. В T_a і T_b видаляється наступна літера, дописується попередня літера (G_2, G_3, G_4, G_5) та в залежності від щойно прочитаної літери ми залишаємося в T_a (T_b), або переходимо в іншу тестову вершину T_b (T_a) відповідно. Тобто слово ми переписуємо із затримкою в один крок і запам'ятовуємо останній прочитаний символ. І, нарешті, якщо в T_a (T_b) потрапили на зірочку, то останній символ буде a (b). Тому в першому випадку дописуємо $a * a$ (G_6, G_7, G_8), а у другому – $b * b$ (G_9, G_{10}, G_{11}) і закінчуємо роботу алгоритму.

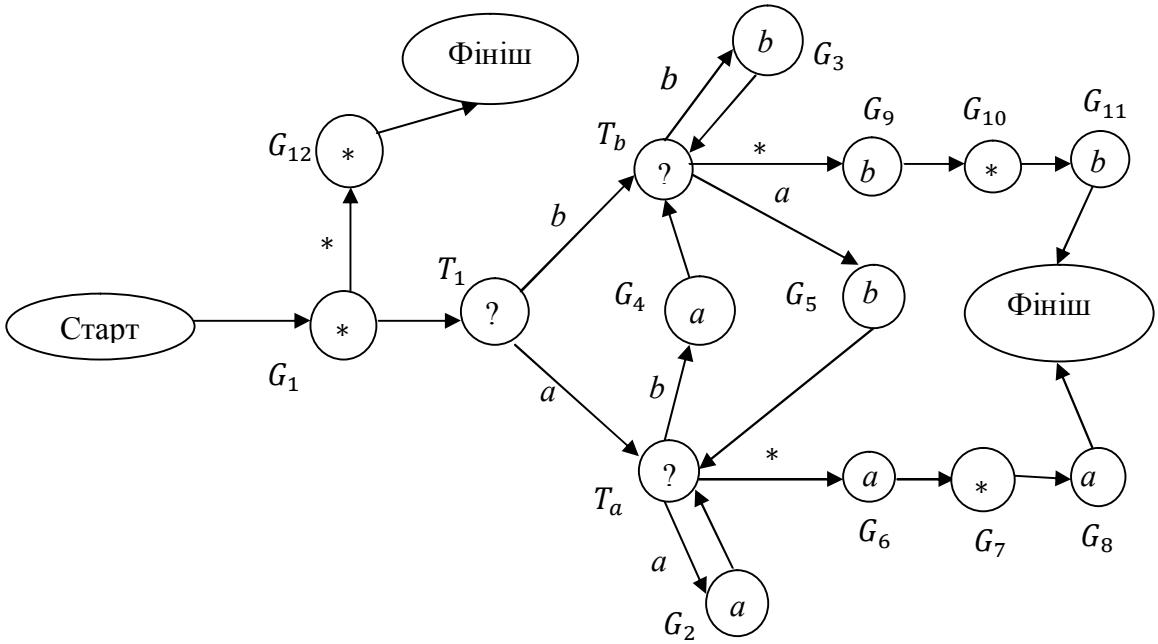


Рис.1.9. БСП для прикладу 1.9

Вправа 1.4. Побудувати БСП, яка:

1. Анулює слово в алфавіті $\{a, b\}$;
2. Видаляє всі літери a, подвоює всі літери b та потроює всі літери c у слові в алфавіті $\{a, b, c\}$;
3. Обчислює $x + 1$ у двійковій системі;
4. Копіює слово в алфавіті $\{a, b\}$;
5. Переводить число з унарної у двійкову систему.

1.5. Універсальна машина Тьюрінга. Проблема зупинки

Задача. Побудувати машину Тьюрінга, яку будемо називати універсальною (УМТ), яка моделювала б роботу заданої МТ на заданому вхідному слові. Для цього потрібно закодувати символи трьох видів: зовнішній алфавіт, внутрішній алфавіт та напрям руху курсору (табл. 1.10).

Табл. 1.10. Кодування символів для УМТ

Символ	Кодування
L	101
R	1001
S	10001
a_1	100001
a_2	10000001
…	……
a_m	$1\underbrace{00\ldots\ldots\ldots00}_{2m+2}1$
q_0	1000001
q_1	100000001
…	……
q_n	$1\underbrace{000\ldots\ldots\ldots00}_{2n+1}1$

Тобто на стрічці в закодованому вигляді розташовується вхідне слово та всі команди, і УМТ сама шукає команду, яку потрібно виконати, та здійснює перетворення вхідного слова.

1.5.1. Проблема зупинки

Задача. Створити машину Тьюрінга на базі УМТ, яка розпізнавала б, чи зупиняється дана МТ на заданому вхідному слові.

Теорема 1.3. Проблема зупинки алгоритмічно нерозв'язна. Тобто таку машину Тьюрінга побудувати неможливо.

Доведення.

Нехай існує машина Тьюрінга T_0, яка розпізнає, чи зупиняється дана машина T на вхідному слові w.

Позначимо через *c(T)* та $\mathrm{c}(w)$ код машини T та код слова w відповідно.

Тобто маємо таку машину

$$T_0\big(\mathrm{c}(T), c(w)\big) = \begin{cases} 1, & \text{коли } T \text{ зупиняється на } w; \\ 0, & \text{коли } T \text{ не зупиняється на } w. \end{cases}$$

Використовуючи МТ T_0 будуємо T_1

$$T_1\big(\mathrm{c}(T), c(w)\big) = \begin{cases} \text{не зупиняється}, & \text{якщо } T \text{ зупиняється на } w; \\ 0 \text{ (зупиняється)}, & \text{якщо } T \text{ не зупиняється на } w. \end{cases}.$$

(Зациклити МТ – елементарна задача).

За допомогою побудованої машини T_1 будуємо МТ T_2, яка з'ясовує, чи зупиняється МТ T на своєму власному коді.

$$T_2\big(\mathrm{c}(T)\big) = \begin{cases} \text{не зупиняється}, & \text{якщо } T \text{ зупиняється на } \mathrm{c}(T); \\ 0, & \text{якщо } T \text{ не зупиняється на } c(T). \end{cases}$$

І, наостанок, замість T підставимо T_2. Маємо

$$T_2\big(\mathrm{c}(T_2)\big) = \begin{cases} \text{не зупиняється}, & \text{якщо } T_2 \text{ зупиняється на } \mathrm{c}(T_2); \\ \text{зупиняється}, & \text{якщо } T_2 \text{ не зупиняється на } c(T_2). \end{cases}$$

Отримали суперечність. Машина зупиняється, коли вона не зупиняється, та зупиняється, коли зупиняється, тому шуканої МТ T_0 не існує.

Кінець доведення.

Висновок. Не можна побудувати універсального налагоджувача, який з'ясовує, чи зупиняється дана процедура на заданих вхідних даних.

Проблема зупинки – перший приклад алгоритмічно нерозв'язної задачі. Тобто алгоритму розв'язування такої задачі не існує взагалі.

З іншими алгоритмічно нерозв'язними проблемами можна ознайомитись, наприклад, в [1, 2].

Розділ 2. Рекурсивні функції

У цьому розділі будемо використовуватися наступні позначення: $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$, $\vec{x}^n = (x_1, x_2, \dots, x_n)$, і розглядаються функції $f^{(n)}: \mathbb{N}_0^n \to \mathbb{N}_0$. Функції визначаються з точністю до фіктивних змінних. При $n = 0$ $f^{(0)} = const$. Верхній індекс у дужках біля функції означає кількість її аргументів.

2.1. Примітивно рекурсивні функції

Означення 2.1. Наступні три функції називають елементарними (базовими):

1. $O(x) = 0$ – анулятор;
2. $P(x) = x + 1$ – інкремент;
3. $I_k^n(\vec{x}^n) = x_k$ ($I_k^n(x_1, x_2, \dots, x_k, \dots, x_n) = x_k$) – проекція $k = \overline{1, n}$.

Всі ці функції вважаються примітивно рекурсивними.

Означення 2.2. Функція $f^{(k)}$ отримана за допомогою суперпозиції функцій $g^{(n)}$, $h_1^{(k)}, h_2^{(k)}, \dots, h_n^{(k)}$, якщо

$$f(\vec{x}^k) = g\left(h_1(\vec{x}^k), h_2(\vec{x}^k), \dots, h_n(\vec{x}^k)\right).$$

Далі введено поняття схеми примітивної рекурсії.

Означення 2.3. Функція $f^{(n+1)}$, $(n \in \mathbb{N}_0)$ отримується за допомогою схеми примітивної рекурсії з функцій $g^{(n)}$ та $h^{(n+2)}$, якщо:

1. $f(\vec{x}^n, 0) = g(\vec{x}^n)$ – база рекурсії;
2. $f(\vec{x}^n, y+1) = h(\vec{x}^n, y, f(\vec{x}^n, y))$ – крок рекурсії.

Тут вектор $\vec{x}^n$ фіксований.

У випадку $n = 0$ $f^{(1)}$ отримується за допомогою схеми примітивної рекурсії з функцій $g^{(0)} = c = const$ та $h^{(2)}$, якщо:

1. $f(0) = c$ – база рекурсії;
2. $f(y+1) = h\big(y, f(y)\big)$ – крок рекурсії.

Означення 2.4. Функція f називається примітивно рекурсивною (ПРФ), якщо:

1. Функція f є однією з базових функцій;
2. f отримується з базових функцій за допомогою скінченної кількості разів використання суперпозицій та схеми примітивної рекурсії.

Приклад 2.1. Покажемо, що константа $f^{(0)} = n$ – ПРФ.

$$1 = P\big(O(x)\big),\ n = \underbrace{P(P(\dots P}_{n \text{ разів}}(O(x)\dots))).$$

Бачимо, що константа – ПРФ, бо застосовувалася скінченна кількість разів суперпозиція інкременту та анулятора.

Приклад 2.2. Довести, що $f(x,y) = x + y$ – ПРФ.

Застосуємо рекурсію по другому аргументу (y)

1. База рекурсії: $f(x,0) = x + 0 = x = I_1^1(x) = g(x)$ – ПРФ.
2. Крок рекурсії: $f(x, y+1) = x + (y+1) = (x+y) + 1 = P(x+y) = = P\big(f(x,y)\big) = h\big(x, y, f(x,y)\big)$.

Тобто бачимо, що функція $f(x,y) = x + y$ отримується за допомогою схеми примітивної рекурсії з функцій

$$g(x) = x = I_1^1(x) \text{ та } h(x, y, z) = P(z) = P\left(I_3^3(x, y, z)\right),$$

які є примітивно рекурсивними, а тому і f буде примітивно рекурсивною.

Приклад 2.3. Довести, що $f(x, y) = x \cdot y$ – ПРФ.

Знову застосуємо рекурсію по другому аргументу.

1. База рекурсії: $f(x, 0) = x \cdot 0 = 0 = O(x) = g(x)$.
2. Крок рекурсії: $f(x, y + 1) = x(y + 1) = xy + x = f(x, y) + x = = h(x, y, f(x, y))$.

Тобто бачимо, що функція $f(x, y) = x \cdot y$ отримується за допомогою схеми примітивної рекурсії з

$$g(x) = O(x) \text{ та } \quad h(x, y, z) = z + x = I_3^3(x, y, z) + + I_1^3(x, y, z),$$

які є ПРФ, ($x + y$ – ПРФ, див. приклад 2.2) і тому сама є ПРФ.

Приклад 2.4. Довести, що $f(x) = x!$ – ПРФ.

1. База рекурсії: $f(x) = 0! = 1 = const$.
2. Крок рекурсії: $f(x + 1) = (x + 1)! = x! \cdot (x + 1) = f(x) \cdot P(x) = = h(x, f(x))$.

Бачимо, що функція $f(x) = x!$ отримується за допомогою схеми примітивної рекурсії з примітивно рекурсивних функцій

$$g = 1 \text{ та } h(x, z) = z \cdot P(x) = I_2^2(x, z) \cdot P\left(I_1^2(x, z)\right)$$

($x \cdot y$ – ПРФ, див. приклад 2.3), і тому сама є ПРФ.

Вправа 2.1. Довести примітивну рекурсивність функцій:

1) $f(x) = x + n$; 2) $f(x, y) = x^y$; 3) $f(x) = x^x$;

4) $sg(x) = \begin{cases} 0, & x = 0 \\ 1, & x > 0 \end{cases}$; 5) $\overline{sg}(x) = \begin{cases} 1, & x = 0 \\ 0, & x > 0 \end{cases}$.

Вправа 2.2. Визначити функції f_1 та f_2, які отримуються за допомогою схеми примітивної рекурсії з функції g_1, h_1 та g_2, h_2 відповідно:

$$g_1(x) = x,\ h_1(x,y,z) = z^x,\ g_2(x) = x,\ h_2(x,y,z) = x^z.$$

2.2. Частково рекурсивні функції

Розглянемо функцію $g^{(n+1)}(n \in \mathbb{N}_0)$**,** яка визначена, можливо, не на усій множині $\mathbb{N}_0^{n+1}$ $(D_g \subset \mathbb{N}_0^{n+1})$, та введемо поняття оператора мінімізації (μ-оператора).

Означення 2.5. Функція $f^{(n)}$ отримується за допомогою μ-оператора з функції $g^{(n+1)}$ $(f(\vec{x}^n) = \mu_y[g(\vec{x}^n, y) = 0])$, якщо:

1. f визначена для таких $\vec{x}^n$, коли знайдеться y такий, що:
   а) $\forall z \leq y - 1\ \ g(\vec{x}^n, z) > 0\ (g - \text{визначена})$;
   б) $g(\vec{x}^n, y) = 0$;
2. $f(\vec{x}^n) = min\{\, y \mid g(\vec{x}, y) = 0\}$.

Тобто оператор мінімізації (μ-оператор) повертає найменше значення аргументу, при якому функція дорівнює 0.

Означення 2.6. Функція називається частково рекурсивною (ЧРФ), якщо:

1. Функція f є однією з базових функцій;
2. f отримується з базових функцій за допомогою скінченної кількості разів використання суперпозиції, схеми примітивної рекурсії та μ-оператора.

Приклад 2.5. Функція, яка не визначена в жодній точці – частково рекурсивна.

$$f(x) = \mu_y[x + y + 1 = 0]$$

Для будь-якого $x \in \mathbb{N}_0$ не знайдеться y, що $x + y + 1 = 0$.

Приклад 2.6. Функція $f(x,y) = \begin{cases} x - y & \text{при } x \geq y \\ \text{не визначена} & \text{при } x < y \end{cases}$ – ЧРФ, бо

$$x - y = \mu_z[|(z + y) - x| = 0], \qquad D_f = \{(x,y) | x \geq y\}.$$

Примітивна рекурсивність функції $f(x,y) = |x - y|$ буде доведена на практичних заняттях.

Приклад 2.7. $f(x) = \begin{cases} \log_2 x & \text{при } x = 2^k \text{ для деякого } k \in \mathbb{N}_0 \\ \text{не визначена} & \text{в інших випадках} \end{cases}$ – ЧРФ, бо

$$\log_2 x = \mu_y\,[|2^y - x| = 0], \qquad D_f = \{x = 2^k, k \in \mathbb{N}_0\}.$$

Узагальнення. μ-оператор можна розглядати і в такому вигляді:

$f(\vec{x}^n) = \mu_y[g_1(\vec{x}^n, y)\ R\ g_2(\vec{x}^n, y)]$, де замість R може стояти деяке відношення на $(\mathbb{N}_0^{n+1})^2$.

Частіше за все розглядаються такі відношення: «=», «≠», «<», «>», «≤». «≥». Усі ці випадки зводяться до стандартного (див. означення 2.5).

Якщо узагальнювати далі, то під знаком μ-оператора можна розглядати довільний $(n+1)$-арний предикат P (характеристична властивість), який повністю визначається своєю характеристичною функцією $\chi_p(\vec{x}^{n+1}) = \begin{cases} 1, & \text{коли } P(\vec{x}^{n+1}); \\ 0, & \text{коли } \neg P(\vec{x}^{n+1}). \end{cases}$

Предикат P називається примітивно (частково) рекурсивним, якщо його характеристична функція χ_p є ПРФ (ЧРФ).

Загальний вигляд μ-оператора виглядає так: $f(\vec{x}^n) = \mu_y\,[P(\vec{x}^n, y)]$.

Означення 2.7. Функція називається загально рекурсивною (ЗРФ), якщо вона є ЧРФ і всюди визначеною.

Приклад 2.8. Довести, що функція $f(x) = \lceil\sqrt{x}\rceil$ (ціла частина $\sqrt{x}$) є ЗРФ.

За означенням $[x] = max\{y \in \mathbb{Z} | y \leq x\}$, тому у випадку

$$\lceil\sqrt{x}\rceil = max\{y \in \mathbb{N}_0 | y \leq \sqrt{x}\} = \ = max\{y \in \mathbb{N}_0 | y^2 \leq x\} =$$
$$= min\{y \in \mathbb{N}_0 | (y+1)^2 > x\} = \mu[(y+1)^2 > x].$$

Функція $\lceil\sqrt{x}\rceil$ визначена для всіх x (для кожного x можна підібрати y, щоб задовольнялася умова $(y+1)^2 > x$), а тому вона є ЗРФ.

Також розглядається поняття обмеженого μ-оператора.

Означення 2.8. μ-оператор називається обмеженим, якщо додатково виконуються дві умови:

1. μ-оператор всюди визначений;
2. $\forall\ \vec{x}^n\ f(\vec{x}^n) \leq h(\vec{x}^n)$ (шуканий y для даного $\vec{x}^n$ не перевищує $h(\vec{x}^n)$).

Записувати це будемо так $f(\vec{x}^n) = \mu_y[g(\vec{x}^n, y) = 0] \leq h(\vec{x}^n)$. Виявляється, що якщо функції g та h – ПРФ, то і f – ПРФ (буде доведено на практичних заняттях).

Тому функція $f(x) = \lceil\sqrt{x}\rceil$ (приклад 2.8) є ПРФ, бо $x, (y+1)^2$ – ПРФ і $\lceil\sqrt{x}\rceil \leq x$, $\forall\ x \in \mathbb{N}_0$.

2.3. Рекурсивність та обчислюваність

Означення 2.9. Функція $f: \mathbb{N}_0^n \to \mathbb{N}_0$ називається ефективно обчислюваною (обчислюваною за Тьюрінгом), якщо існує ефективна процедура (та сама машина Тьюрінга), яка приймає $\vec{x}^n$ і повертає $f(x)$. У випадку, коли функція не визначена, машина Тьюрінга не зупиняється. Тут $\vec{x}^n$ та $f(x)$ можуть записуватися, наприклад, в унарній системі.

Теорема 2.1. Функція $f: \mathbb{N}_0^n \to \mathbb{N}_0$ є чисельно ефективно обчислюваною тоді і тільки тоді, коли вона є частково рекурсивною.

Доведення даної теореми опустимо, лише відмітимо: щоб довести, що будь-яка ЧРФ є ефективно обчислювальною, потрібно вміти будувати машину Тьюрінга для базових функцій $O(x)$, $P(x)$, $I_k^n(\vec{x}^n)$, а також реалізовувати за допомогою машини Тьюрінга суперпозицію функцій, схему примітивної рекурсії та μ-оператор.

У підсумку отримуємо, що наступні твердження еквівалентні для $f: \mathbb{N}_0^n \to \mathbb{N}_0$.

1. f обчислюється за допомогою машини Тьюрінга;
2. f обчислюється за допомогою нормального алгоритму Маркова;
3. f обчислюється за допомогою блок-схеми Поста;
4. f – частково рекурсивна функція.

Теза Тьюрінга-Черча[3]: Процедура є ефективною, тобто алгоритмом тоді і тільки тоді, коли вона реалізується за допомогою Машини Тьюрінга.

[3] Алонзо Черч (1903 – 1995) – видатний американський математик, логік, здійснив ряд фундаментальних відкриттів у символічній логіці та теорії обчислювальності.

Числова функція ефективно обчислювана тоді і тільки тоді, коли вона є частково рекурсивною.

Фактично дане твердження дає визначення алгоритму.

Подібні міркування можна проводити не тільки для числових функцій.

2.4. Поняття про словникові функції

Розглянемо скінченний алфавіт $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$ і функції $f: (\mathcal{A}^*)^n \to \mathcal{A}^*$ ($\mathcal{A}^*$ – множина всіх слів над алфавітом $\mathcal{A}$, див. параграф 1.3).

Означення 2.9. Базовими функціями називаються:

1. $O(x) = \varepsilon$ – анулятор;
2. $P_i(x) = xa_i, \quad i = \overline{1,n}$ – інкременти (до слова x дописується літера a_i);
3. $I_k^n(\vec{x}^n) = x_k$ – проекція.

Суперпозиція функцій визначається аналогічно числовому випадку, а схема примітивної рекурсії – наступним чином:

Означення 2.10. $f^{(n+1)}(n \in \mathbb{N}_0)$ отримується за допомогою схеми примітивної рекурсії з функцій $g^{(n)}$ та $h_i^{(n+2)}$ $i = \overline{1,n}$, якщо:

1. $f(\vec{x}^n, \varepsilon) = g(\vec{x}^n)$ – база рекурсії;
2. $f(\vec{x}^n, ya_i) = h_i\big(\vec{x}^n, y, f(\vec{x}^n, y)\big)$ $i = \overline{1,n}$ – крок рекурсії.

На множині слів з $\mathcal{A}^*$ вводиться відношення лінійного порядку (наприклад, слова впорядковуються за довжиною, а серед слів однакової довжини вводиться лексикографічний порядок).

Тоді означення μ-оператора буде аналогічним числовому випадку. Також аналогічним чином визначаються поняття ПРФ (ЧРФ, ЗРФ), і справедливою буде теорема, аналогічна теоремі 2.1.

Приклад 2.9. Довести примітивну рекурсивність функції $f(x,y) = xy$ (конкатенція, слово y дописується справа від x).

1. База рекурсії: $f(x,\varepsilon) = x = I_1^1(x) = g(x)$ – ПРФ;
2. Крок рекурсії:

$f(x, ya_i) = x(ya_i) = (xy)a_i = P_i\big(f(x,y)\big) = h_i\big(x,y,f(x,y)\big)\ \ i = \overline{1,n}.$

Тобто бачимо, що функція $f(x,y) = xy$ отримується за допомогою схеми примітивної рекурсії з функцій

$g(x) = x = I_1^1(x)$ та $h_i(x,y,z) = P_i(z) = P_i(I_3^3(x,y,z))$,

які є примітивно рекурсивними, а тому сама f буде примітивно рекурсивною.

Розділ 3. Формальні логічні теорії

3.1. Поняття логічної теорії

Теорія називається змістовною, якщо існує деяка інтерпретація об'єктів, операцій, символів теорії. Зокрема алгебра висловлень (див., наприклад, [1, 6]) – змістовна теорія, бо кожен символ в ній інтерпретується як просте висловлення, яке може бути правдивим або хибним. На відміну від змістовних теорій, формальні теорії є множиною символів та співвідношень між символами, яким не надається жодного змістовного сенсу. Вирази та формули формальної теорії є «чистими» схемами міркувань, яким можна надавати різноманітний сенс, тобто будувати різні моделі теорії.

Означення 3.1. Формальна логічна теорія будується за наступними правилами:

1. Задається алфавіт теорії – зліченна множина символів.
2. З множини символів алфавіту будуються скінченні послідовності за певними правилами. Таким чином отримуємо формули формальної теорії (частіше за все задаються рекурсивно).
3. З множини формул виокремлюється підмножина формул, яку називатимемо множиною (системою) аксіом даної теорії.
4. Між формулами теорії визначаються відношення – правила виведення (дозволяють від одних формул переходити до інших).

Ключовим поняттям даного розділу є поняття формального виведення.

Означення 3.2. Формальним виведенням формули B з множини гіпотез Г називається скінченна послідовність формул $A_1, A_2, \dots, A_n$, кожна

з яких може бути або аксіомою, або гіпотезою з множини Γ, або отриманою з попередніх за допомогою одного з правил виведення. Позначати формальне виведення будемо $\Gamma \vdash_T B$ («⊢» – знак формального виведення, «T» – конкретизується, яка формальна теорія використовується в разі необхідності).

Означення 3.3. Формула B називається теоремою в формальній теорії T, якщо можна побудувати формальне виведення формули B з порожньої множини гіпотез ($\vdash_T B$). Інакше кажучи, теорема – формула, отримана з множини аксіом за допомогою правил виведення.

Наведемо деякі властивості формального виведення:

1. Якщо $\Gamma \subset \Delta$ і $\Gamma \vdash B$, то $\Delta \vdash B$.
2. $\Gamma \vdash B$ тоді і тільки тоді, коли $\exists \Delta \subset \Gamma$ така, що $\Delta \vdash B$.
3. Якщо $\Delta \vdash A$ і $\Gamma \vdash B_i$, $\forall B_i \subset \Delta$, то $\Gamma \subset \Delta$.

3.2. Числення висловлень. Формальна теорія L

3.2.1. Означення формальної теорії L

Формальна теорія L визначається наступним чином:

Означення 3.4.

1. Алфавіт – пропозиційні літери, які будемо позначати великими літерами латинської абетки з індексами або без індексів ($A, B, C, \ldots$), логічні зв'язки ¬, → та допоміжні символи «(», «,», «)».
2. Формула в формальній теорії визначається так:
   а) Будь-яка пропозиційна літера є формулою;
   б) Якщо A та B формули, то формулами є $(\neg A)$ та $(A \to B)$;

в) Інших формул немає.

3. У формальній теорії L визначена нескінченна множина аксіом, яка задається трьома схемами аксіом:

A1: $A \to (B \to A)$;

A2: $\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)$;

A3: $(\neg B \to \neg A) \to \big((\neg B \to A) \to B\big)$.

Конкретна аксіома отримується підстановкою формули замість пропозиційної літери.

4. У теорії L визначено єдине правило виведення MP (Modus Ponens) $A,\ A \to B \vdash B$.

Надалі для скорочення запису зовнішні дужки будемо опускати і вважаємо, що $\neg$ має пріоритет над $\to$. Крім того, для скорочення запису формул вводяться наступні метаозначення:

MO1: $\neg A \to B = A \vee B$;

MO2: $\neg(A \to \neg B) = A \wedge B$;

MO3: $(A \to B) \wedge (B \to A) = A \leftrightarrow B$.

3.2.2. Доведення та виведення в формальній теорії L

Доведемо деякі теореми та побудуємо формальні виведення в формальній теорії L.

Теорема 1. $\vdash A \to A$.

1.	$\big(A \to \big((A \to A) \to A\big)\big) \to \big((A \to (A \to A)) \to (A \to A)\big)$	A2
2.	$A \to ((A \to A) \to A)$	A1
3.	$\big(A \to (A \to A)\big) \to (A \to A)$	MP (1, 2)
4.	$A \to (A \to A)$	A1
5.	$A \to A$	MP (3, 4)

Теорема 2. $\vdash (\neg A \to A) \to A$.

1.	$(\neg A \to \neg A) \to \big((\neg A \to A) \to A\big)$	A3
2.	$\neg A \to \neg A$	T1
3.	$(\neg A \to A) \to A$	MP (1, 2)

При доведенні теореми 2 у пункті 2 ми посилаємося на доведену раніше теорему 1. Це означає, що замість 2 слід включити всі пункти доведення теореми 1.

В1. Правило силогізму $A \to B, B \to C \vdash A \to C$.

1.	$A \to B$	Г1
2.	$B \to C$	Г2
3.	$\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)$	A2
4.	$(B \to C) \to \big(A \to (B \to C)\big)$	A1
5.	$A \to (B \to C)$	MP (2, 4)
6.	$(A \to B) \to (A \to C)$	MP (5, 3)
7.	$A \to C$	MP (1, 6)

В2. Правило видалення середньої посилки $A \to (B \to C), B \vdash A \to C$.

1.	$A \to (B \to C)$	Г1
2.	B	Г2
3.	$\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)$	A2

4.	$(A \to B) \to (A \to C)$	MP (1, 3)
5.	$B \to (A \to B)$	A1
6.	$B \to (A \to C)$	B1 (5, 4)
7.	$A \to C$	MP (2, 6)

B3. Правило видалення крайньої посилки $(A \to B) \to C \vdash B \to C$.

1.	$(A \to B) \to C$	Г1
2.	$B \to (A \to B)$	A1
3.	$B \to C$	B1 (2, 1)

Теорема 3. $\vdash \neg\neg A \to A$ (видалення подвійного заперечення).

1.	$(\neg A \to \neg\neg\neg A) \to \big((\neg A \to \neg A) \to A\big)$	A3
2.	$\neg A \to \neg A$	T1
3.	$(\neg A \to \neg\neg\neg A) \to A$	B2 (1, 2)
4.	$\neg\neg A \to A$	B3 (3)

Теорема 4. $\vdash A \to \neg\neg A$ (введення подвійного заперечення).

1.	$(\neg\neg\neg\neg A \to \neg A) \to \big((\neg\neg\neg\neg A \to A) \to \neg\neg A\big)$	A3
2.	$\neg\neg\neg\neg A \to \neg A$	T3
3.	$(\neg\neg\neg\neg A \to A) \to \neg\neg A$	MP (1, 2)
4.	$A \to \neg\neg A$	B3 (3)

Зауваження 3.1. У всіх вище наведених теоремах та правилах виведення, а також у тих, що будуть пізніше, замість кожної пропозиційної літери може стояти будь-яка формула.

Для розв’язання задач на побудову формального виведення слід використовувати ті аксіоми та раніше доведені теореми та правила

виведення, які в лівій частині містять гіпотези та отримані проміжні формули, а в правій – те, що маємо отримати.

Також дуже корисною для формального доведення є метатеорема дедукції (МТД).

Теорема 3.1. (метатеорема дедукції для теорії L),

$$\Gamma, A \vdash B \Leftrightarrow \Gamma \vdash A \to B.$$

Доведення.

Необхідність. Нехай маємо формальне виведення $\Gamma, A \vdash B$ $(B_1, B_2, \dots, B_n = B)$.

Потрібно побудувати формальне виведення $\Gamma, A \vdash B$. Застосуємо метод математичної індукції по n.

База індукції. Нехай $n = 1$. Розглянемо три випадки:

а) $\vdash B_1$, тут B_1 може бути аксіомою;

запишемо аксіому із серії А1 $B_1 \to (A \to B_1)$ і отримаємо $\vdash A \to B_1 \Rightarrow \Gamma \vdash A \to B_1$, використавши правило МР.

б) $\vdash B_1$, тут B_1 може бути також гіпотезою $B_1 \in \Gamma$. Доводиться аналогічно пункту а).

в) $B_1 = A$, тобто $\vdash A$.

Використаємо теорему 1 $\vdash A \to A \Rightarrow \Gamma \vdash A \to A$.

Припущення індукції. Нехай твердження МТД виконується для всіх $n \leq k$.

Крок індукції. Доведемо, що воно виконується і при $n = k + 1$. Тут можливі чотири випадки. Крім вище написаних випадків а), б), в) може додатися випадок г).

г) Формула B_{k+1} отримана з попередніх формул за допомогою правила MP. Тобто

B_i

$B_j = B_i \to B_{k+1}$ $\qquad (i < j \leq k)$

За припущенням індукції маємо виведення: $\Gamma \vdash A \to B_i$ та $\Gamma \vdash A \to B_j = A \to (B_i \to B_{k+1})$. Запишемо аксіому із схеми А2

$$\big(A \to (B_i \to B_{k+1})\big) \to \big((A \to B_i) \to (A \to B_{k+1})\big).$$

Двічі застосуємо правило MP і отримаємо $\Gamma \vdash A \to B_{k+1}$.

Необхідність доведена.

Достатність. Нехай існує виведення $\Gamma \vdash A \to B$. Доведемо, що формула B виводиться з Γ та A. Нехай виведення формули $A \to B$ має вигляд $B_1, B_2, \dots, B_n = A \to B$. Допишемо до нього $B_{n+1} = A$ і за правилом MP отримаємо $B_{n+2} = B$.

Теорема доведена повністю.

Зауваження 3.2. Сама МТД не є частиною теорії L, оскільки метод математичної індукції, який використовується при її доведенні, не постулюється і не виводиться в формальній теорії L.

Зауваження 3.3. Твердження $\Gamma, A \vdash B \Rightarrow \Gamma \vdash A \to B$ називається прямою МТД, а зворотнє твердження $\Gamma \vdash A \to B \Rightarrow \Gamma, A \vdash B$ – оберненою МТД. Сама обернена МТД часто застосовується для побудови формальних виведень. Зручність використання цього твердження полягає в тому, що часто буває вигідно записати в множину гіпотез як можна більше формул, а доводити доведеться простішу формулу, ніж була до того.

Продовжимо будувати формальні виведення, тепер з використанням МТД.

Теорема 5. $\vdash \neg A \to (A \to B)$ (суперечність).

Двічі застосуємо обернену МТД і отримаємо: $\vdash \neg A \to (A \to B) \Leftrightarrow \neg A \vdash A \to B \Leftrightarrow \neg A, A \vdash B$. Будуємо виведення.

1.	$\neg A$	Г1
2.	A	Г2
3.	$(\neg B \to \neg A) \to \big((\neg B \to A) \to B\big)$	А3
4.	$\neg A \to (\neg B \to \neg A)$	А1
5.	$A \to (\neg B \to A)$	А1
6.	$\neg B \to \neg A$	МР (1, 4)
7.	$\neg B \to A$	МР (2, 6)
8.	$(\neg B \to A) \to B$	МР (3, 6)
9.	B	МР (7, 8)

Теорема 6. $\vdash (\neg A \to \neg B) \to (B \to A)$ (контрапозиція).

Один раз використаємо обернену МТД.

$\vdash (\neg A \to \neg B) \to (B \to A) \Leftrightarrow \neg A \to \neg B \vdash B \to A$.

1.	$\neg A \to \neg B$	Г1
2.	$(\neg A \to \neg B) \to \big((\neg A \to B) \to A\big)$	А3
3.	$(\neg A \to B) \to A$	МР (1, 2)
4.	$B \to A$	В3 (3)

Теорема 7. $\vdash (B \to A) \to (\neg A \to \neg B)$ (контрапозиція).

Один раз використаємо обернену МТД.

$\vdash (B \to A) \to (\neg A \to \neg B) \Leftrightarrow B \to A \vdash \neg A \to \neg B.$

1.	$B \to A$	Г1
2.	$\neg\neg B \to B$	Т3
3.	$A \to \neg\neg A$	Т4
4.	$\neg\neg B \to A$	В1 (1, 2)
5.	$\neg\neg B \to \neg\neg A$	В1 (3, 4)
6.	$(\neg\neg B \to \neg\neg A) \to (\neg A \to \neg B)$	Т6
7.	$\neg A \to \neg B$	МР (5, 6)

Вправа 3.1. За допомогою МТД побудувати формальні виведення:

В4 $A \to (B \to C) \vdash B \to (A \to C)$;

В5 $A \vdash (A \to B) \to B$.

Теорема 8. $\vdash A \to \big(\neg B \to \neg(A \to B)\big)$.

$\vdash A \to (\neg B \to \neg(A \to B)) \overset{\text{МТД}}{\Longleftrightarrow} A \vdash \neg B \to \neg(A \to B).$

1.	A	Г1
2.	$(A \to B) \to B$	В5 (1)
3.	$((A \to B) \to B) \to (\neg B \to \neg(A \to B))$	Т7
4.	$\neg B \to \neg(A \to B)$	МР (2, 3)

Теорема 9. $\vdash (A \to B) \to \big((\neg A \to B) \to B\big)$.

$$\vdash (A \to B) \to \big((\neg A \to B) \to B\big) \overset{\text{МТД}}{\Longleftrightarrow} A \to B \vdash \big((\neg A \to B) \to B\big) \overset{\text{МТД}}{\Longleftrightarrow}$$
$$\overset{\text{МТД}}{\Longleftrightarrow} A \to B, \neg A \to B \vdash B.$$

1.	$A \to B$	Г1
2.	$\neg A \to B$	Г2

3.	$(A \to B) \to (\neg B \to \neg A)$	T6
4.	$\neg B \to \neg A$	MP (1, 3)
5.	$\neg B \to B$	B1 (4, 2)
6.	$(\neg B \to B) \to B$	T2
7.	B	MP (5, 6)

3.2.3. Правила введення та видалення зв'язок

При побудові формальних виведень можна використовувати довільні теореми та правила виводу, доведені раніше, а також наступні правила введення та видалення зв'язок (табл. 3.1).

Таблиця 3.1. Правила введення та видалення зв'язок

Зв'язка	Введення	Видалення
$\to$	$\Gamma, A \vdash B \Rightarrow \Gamma \vdash A \to B$ (МТД)	$A, A \to B \vdash B$ (MP) $\Gamma_1 \vdash A, \Gamma_2 \vdash A \to \text{B} \Rightarrow \Gamma_1, \Gamma_2 \vdash B$
$\neg$	$\dfrac{\Gamma, A \vdash B,\ \Gamma, A \vdash \neg B}{\Gamma \vdash \neg A}$	$\neg\neg A \vdash A$ (видалення подвійного заперечення)
$\wedge$	$\dfrac{\Gamma_1 \vdash A;\ \Gamma_2 \vdash B}{\Gamma_1, \Gamma_2 \vdash A \wedge B}$	$\Gamma \vdash A \wedge B \Rightarrow \Gamma \vdash A$ $\Gamma \vdash A \wedge B \Rightarrow \Gamma \vdash B$
$\vee$	$\Gamma \vdash A \Rightarrow \Gamma \vdash A \vee B$ $\Gamma \vdash B \Rightarrow \Gamma \vdash A \vee B$	$\dfrac{\Gamma_1 \vdash A \vee B;\ \Gamma_2, A \vdash C;\ \Gamma_3, B \vdash C}{\Gamma_1, \Gamma_2, \Gamma_3 \vdash C}$
$\leftrightarrow$	$A \to B, B \to A \vdash A \leftrightarrow B$	$A \leftrightarrow B \vdash A \to B$ $A \leftrightarrow B \vdash B \to A$

Доведемо правило видалення диз'юнкції:

$$\Gamma_1 \vdash A \vee B;\ \Gamma_2, A \vdash C;\ \Gamma_3, B \vdash C \Rightarrow \Gamma_1, \Gamma_2, \Gamma_3 \vdash C.$$

Дана задача зводиться до наступної: $A \vee B = \neg A \to B,\ A \to C,\ B \to C \vdash C$. Тут використовується пряма МТД.

1.	$\neg A \to B$	Г1
2.	$A \to C$	Г2
3.	$B \to C$	Г3
4.	$\neg A \to C$	В1 (1, 3)
5.	$(A \to C) \to ((\neg A \to C) \to C)$	Т9
6.	$(\neg A \to C) \to$ C	MP (2, 5)
7.	C	MP (4, 6)

Вправа 3.2. Довести інші твердження, наведені в таблиці 3.1.

Проілюструємо на прикладі використання різних правил введення та видалення зв'язок.

Приклад 3.1. Довести теорему в формальній теорії L

$$\vdash \big(A \vee (B \wedge C)\big) \to \big((A \vee B) \wedge (A \vee C)\big)$$

Використовуючи МТД, отримаємо еквівалентну задачу:

$$A \vee (B \wedge C) \vdash (A \vee B) \wedge (A \vee C).$$

Також позначимо через $Q^+(Q^-)$ правило введення (видалення) логічної зв'язки Q.

1.	$A \vee (B \wedge C)$	Г1
2.	$A \vdash A \vee B$	$\vee^+$
3.	$A \vdash A \vee C$	$\vee^+$
4.	$B \wedge C \vdash B$	$\wedge^-$
5.	$B \wedge C \vdash C$	$\wedge^-$
6.	$B \vdash A \vee B$	$\vee^+$
7.	$C \vdash A \vee C$	$\vee^+$
8.	$B \wedge C \vdash A \vee B$	(4, 6 та МТД)
9.	$B \wedge C \vdash A \vee C$	(5, 7 та МТД)
10.	$A \vee B$	$\vee^- (1, 2, 8)$
11.	$A \vee C$	$\vee^- (1, 3, 9)$
12.	$(A \vee B) \wedge (A \vee C)$	$\wedge^+ (10, 11)$

3.2.4.Адекватність формальної теорії L та алгебри висловлень.

Для досліджень властивостей формальної теорії зазвичай будується її модель (інтерпретація).

Означення 3.5. Кажуть, що змістовна теорія T_1 є моделлю формальної теорії T, якщо:

1. Між множинами формул обох теорій можна побудувати взаємно однозначну відповідність.
2. Кожна теорема в теорії T буде тавтологією (тотожно правдивою, виділеною формулою) в теорії T_1.

У теорії T_1 можуть існувати свої тавтології, які не є теоремами в T.

Означення 3.6. Якщо множина теорем теорії T співпадає з множиною тавтологій теорії T_1, то T називаються адекватною T_1,.

Покажемо адекватність формальної теорії L та алгебри висловлень.

Зрозуміло, що кожній формулі теорії L можна співставити формулу в алгебрі висловлень і навпаки.

Слід згадати МО1: $A \vee B = \neg A \rightarrow B$, МО2: $A \wedge B = \neg(A \rightarrow \neg B)$.

Далі нам потрібно довести, що кожна формула буде теоремою в теорії L тоді і тільки тоді, коли вона є тавтологією в алгебрі висловлень.

Теорема 3.2. Кожна теорема в формальній теорії L є тавтологією в алгебрі висловлень.

Доведення. Нагадаємо, що теорема в формальній теорії L – формула, яку можна довести безпосередньо з аксіом серій А1, А2, А3 та правила виведення Modus Ponens. Тому для доведення твердження теореми потрібно показати, що дані схеми аксіом є тавтологіями, а також те, що правило МР зберігає тавтології.

Вправа 3.3. За допомогою таблиць правдивості переконатися в справедливості твердження теореми. 3.2.

Кінець доведення теореми 3.2.

Теорема 3.3. Кожна тавтологія в алгебрі висловлень є теоремою в формальній теорії L.

Для доведення даної теореми нам знадобиться допоміжне твердження.

Лема 3.1. Нехай формула B залежить від пропозиційних літер

$$A_1,\ A_2, \dots, A_n, B = B(A_1,\ A_2, \dots, A_n), n \geq 0.$$

Тоді по кожному рядку таблиці правдивості формули B (табл. 3.2)

Таблиця 3.2.

A_1	A_2	…	A_n	B
α_1	α_2	…	α_n	β

можна побудувати формальне виведення $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash \tilde{B}$,

де $\tilde{A}_i = \begin{cases} A_i, \text{ коли } \alpha_i = 1; \\ \neg A_i, \text{ коли } \alpha_i = 0, \end{cases} i = \overline{1,n}, B = \begin{cases} B, \text{ коли } \beta = 1; \\ \neg B, \text{ коли } \beta = 0. \end{cases}$

Перед тим, як почати доводити лему 3.1, розглянемо приклад її застосування.

Приклад 3.2. Нехай $B = A_1 \to A_2$.

Таблиця правдивості імплікації виглядає наступним чином:

A_1	A_2	$A_1 \to A_2$
0	0	1
0	1	1
1	0	0
1	1	1

Тому мають існувати формальні виведення:

1. $\neg A_1, \neg A_2 \vdash A_1 \to A_2$;
2. $\neg A_1, A_2 \vdash A_1 \to A_2$;
3. $A_1, \neg A_2 \vdash \neg(A_1 \to A_2)$;
4. $A_1, A_2 \vdash A_1 \to A_2$.

Дійсно, пункти 1 та 2 справджуються, бо внаслідок теореми 5 та МТД $\vdash \neg A_1 \to (A_1 \to A_2) \Leftrightarrow \neg A_1 \vdash A_1 \to A_2$.

Пункт 4. А1 та МТД $\vdash A_2 \to (A_1 \to A_2) \Leftrightarrow A_2 \vdash A_1 \to A_2$.

Пункт 3. Використовується теорема 8 та МТД.

$$\vdash A_1 \to \big(\neg A_2 \to \neg(A_1 \to A_2)\big) \Leftrightarrow A_1, \neg A_2 \vdash \neg(A_1 \to A_2).$$

Доведення леми 3.1. Застосуємо метод математичної індукції по кількості логічних зв'язок, які входять у формулу B.

A_i	A_i
0	0
1	1

База індукції.

$m = 0, B = A_i$.

Тоді очевидно, що $\neg A_i \vdash \neg A_i$ $A_i \vdash A_i$.

Припущення індукції. Нехай для формули B, яка містить не більше, ніж k ($m \leq k$) логічних зв'язок, твердження леми виконується.

Крок індукції. Нехай формула має $k + 1$ логічну зв'язку ($m = k + 1$).

Розглянемо два випадки:

1) $B = \neg B_1$;
2) $B = B_1 \to B_2$.

У першому випадку за припущенням індукції можна побудувати формальне виведення

$\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash \tilde{B}_1$. Якщо $\tilde{B}_1 = B_1$, тобто $\beta_1 = 1$, тоді $\beta = 0$ і будується формальне виведення $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash \neg B = \neg\neg B_1$.Це дійсно так, бо $B_1 \vdash \neg\neg B_1$ (Т4 та МТД). Якщо ж $\tilde{B}_1 = \neg B_1$, тоді $\beta_1 = 0$, а $\beta = 1$. У нас є виведення $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash \neg B_1$. Оскільки $\neg B_1 = B$, то матимемо виведення $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash B$.

У другому випадку за припущенням індукції можна побудувати формальні виведення: $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash \tilde{B}_1$ та $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash \tilde{B}_2$.

Цей випадок розділяється ще на три окремі випадки:

а) $\tilde{B}_1 = \neg B_1$;

б) $\tilde{B}_2 = B_2$;

в) $\tilde{B}_1 = B_1, \tilde{B}_2 = \neg B_2$.

У випадках а) та б) $\beta = 1$, тобто потрібно будувати виведення $A_1, A_2, \dots, A_n \vdash B_1 \to B_2$. Це можна зробити, бо

а) $\vdash \neg B_1 \to (B_1 \to B_2) \Leftrightarrow \neg B_1 \vdash B_1 \to B_2$ (Т5, МТД).

б) $\vdash B_2 \to (B_1 \to B_2) \Leftrightarrow B_2 \vdash B_1 \to B_2$.

У випадку в) $\beta = 0$ ($\beta_1 = 1, \beta_2 = 0$), тому виведення

$\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash \neg(B_1 \to B_2)$ слідує з Т8 та МТД.

$\vdash B_1 \to \left(\neg B_2 \to \neg(B_1 \to B_2)\right) \Leftrightarrow B_1, \neg B_2 \vdash \neg(B_1 \to B_2)$.

Лема доведена.

Повернемося до доведення теореми.

Доведення теореми 3.3. Нехай формула $B = B(A_1, A_2, \dots, A_n)$, $n \geq 0$ є тавтологією в алгебрі висловлень. Покажемо, що формула B є теоремою в теорії L ($\vdash B$). Кожен рядок таблиці правдивості формули B має вигляд:

A_1	A_2	…	A_n	B
α_1	α_2	…	α_1	1

І тому внаслідок леми 3.1 маємо 2^n формальних виведень $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \vdash B$. Розглянемо два виведення, які відрізняються лише $\tilde{A}_n$

$$(\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_{n-1}, A_n \vdash B \text{ та } \tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_{n-1}, \neg A_n \vdash B) \overset{\text{МТД}}{\Longleftrightarrow}$$
$$\overset{\text{МТД}}{\Longleftrightarrow} (\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_{n-1}, \vdash A_n \to B \text{ та } \tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_{n-1}, \vdash \neg A_n \to B).$$

Згадаємо Т9, $\vdash (A_n \to B) \to \left((\neg A_n \to B) \to B\right) \overset{\text{МТД}}{\Longleftrightarrow} A_n \to B, \neg A_n \to B \vdash B$.

Таким чином отримаємо $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_{n-1}, \vdash B$, тобто маємо таке ж виведення, але без $\tilde{A}_n$. Продовжуючи цю процедуру, послідовно виключаємо $\tilde{A}_{n-1}$, $\tilde{A}_{n-2}, \dots, \tilde{A}_1$ і отримаємо $\vdash B$.

Теорема доведена повністю.

Формальна теорія L ще називається численням висловлень.

3.2.5. Інші властивості формальної теорії L

Означення 3.7. Формальна теорія називається *розв'язною*, якщо існує ефективна процедура, яка за скінченну кількість кроків дозволяє з'ясовувати, чи є формула теоремою, чи ні.

Твердження 3.1. Формальна теорія L розв'язна, бо за теоремами 3.2 та 3.3 перевірку, чи є формула теоремою, можна здійснити за допомогою таблиці правдивості. Формула має бути тавтологією.

Означення 3.8. Формальна теорія називається *суперечливою*, коли в ній можна вивести дві протилежні (суперечливі) формули $(\exists A\ (\vdash A) \wedge (\vdash \neg A))$.

Твердження 3.2. Формальна теорія L несуперечлива, бо не може одна і та сама формула разом із запереченням бути тавтологіями.

Означення 3.9. Формальна теорія називається *повною у широкому сенсі*, якщо будь-яку її формулу можна підтвердити або спростувати $(\forall A\ (\vdash A) \vee (\vdash \neg A))$.

Тобто або сама формула, або її заперечення обов'язково будуть теоремами.

Твердження 3.3. Формальна теорія L неповна в широкому сенсі, бо в алгебрі висловлень існують нейтральні формули, тобто ні сама формула, ні її заперечення не є тавтологіями.

Означення 3.10. Формальна теорія називається *повною у вузькому сенсі*, якщо додавання в систему аксіом будь-якої формули, яку не можна довести, робить її суперечливою.

Твердження 3.4. Формальна теорія L повна у вузькому сенсі.

Доведення. До трьох схем аксіом A1, A2, A3 та правила виводу MP додамо формулу A, яка не є тавтологією. Маємо теорію L': A1, A2, A3,

A та МР. Оскільки A не тавтологія, то кон'юктивна нормальна форма (КНФ) її непорожня, тобто містить хоча б один нетавтологічний диз'юнкт $A = D_1 \wedge D_2 \wedge ... \wedge D_k$. Внаслідок правила видалення кон'юнкції отримаємо $A \vdash D_1$, і нехай

$$D_1 = \tilde{B}_1 \vee \tilde{B}_2 \vee ... \vee \tilde{B}_m, \text{ де } \tilde{B}_i = \begin{cases} B_i \\ \neg B_i \end{cases} i = \overline{1,m}.$$

У диз'юнкті D_1 замінимо всі $\tilde{B}_i$ на B, якщо $\tilde{B}_i = B_i$ і $\tilde{B}_i$ замінимо на $\neg B$, якщо $\tilde{B}_i = \neg B_i$. Тоді з D_1 виводиться B, а тому $A \vdash B$. Потім у диз'юнкті D_1 проведемо іншу заміну. Замінимо всі $\tilde{B}_i$ на $\neg B$, якщо $\tilde{B}_i = B_i$ і $\tilde{B}_i$ на B, якщо $\tilde{B}_i = \neg B_i$. Цього разу ми отримаємо $\neg B$ ($D_1 \vdash \neg B \Longrightarrow A \vdash \neg B$).

Бачимо, що теорія L' є суперечливою, а тому теорія L є повною у вузькому сенсі.

Кінець доведення.

Означення 3.11. Система аксіом формальної теорії називається незалежною, якщо будь-яку з них не можна вивести з інших.

Твердження 3.5. Схеми аксіом А1, А2, А3 формальної теорії L є незалежними.

Доведення. Доведемо, що А1 не залежить від А2 та А3, тобто з А2 та А3 не можна вивести А1. Для цього побудуємо несуперечливу модель в тризначній логіці (на множині {0,1,2}), в якій виконуються А2 та А3, і не виконується А1 (табл. 3.3. 3.4).

Формулу, яка тотожно дорівнює нулю, назвемо виділеною, або тавтологією.

Формула $A \to (B \to A)$ з А1 не тавтологія, бо $1 \to (2 \to 1) = 1 \to 0 = 2$.

Вправа 3.4. Переконатися самостійно, що схеми аксіом А2 та А3 є тавтологіями в даній моделі та правило МР зберігає тавтології. Тому аксіоми із схеми А1 не можна вивести з А2 та А3.

Доведення незалежності А2 та А3 можна знайти, наприклад, в [1].

Кінець доведення.

Таблиця 3.3.

A	$\neg A$
0	1
1	1
2	0

Таблиця 3.4.

A	B	$A \to B$
0	0	0
1	0	2
2	0	0
0	1	2
1	1	2
2	1	0
0	2	2
1	2	0
2	2	0

3.3. Інші формалізації числення висловлень

Крім теорії L існують інші формальні логічні теорії, які є адекватними алгебрі висловлень.

Теорія L_1 (Гільберт, Аккерман).

Основні зв'язки: $\vee, \neg$

МО: $A \to B = \neg A \vee B$.

Схеми аксіом:

А1: $(A \vee A) \to A$;

A2: $A \to (A \vee B)$;

A3: $(A \vee B) \to (B \vee A)$;

A4: $(B \to C) \to \big((A \vee B) \to (A \vee C)\big)$.

Правило виведення MP.

Теорія L_2 (Россер).

Основні зв'язки: $\wedge, \neg$.

МО: $A \to B = \neg(A \wedge \neg B)$.

Схеми аксіом:

A1: $A \to (A \wedge A)$;

A2: $(A \wedge B) \to A$;

A3: $(A \to B) \to \big(\neg(B \wedge C) \to \neg(C \wedge A)\big)$.

Правило виведення MP.

Про інші формальні теорії, адекватні алгебрі висловлень, можна дізнатися, наприклад, в [1].

Розділ 4. Алгебра предикатів

4.1. Основні поняття алгебри предикатів

4.1.1. Визначення предикатів. Поняття квантора

Існують такі види логічних міркувань, які неможна обґрунтувати в рамках алгебри висловлень. Прикладами таких міркувань можуть слугувати наступні:

1. Кожен друг Петра є другом Василя. Павло не є другом Василя. Отже, Павло не є другом Петра.
2. Всі люди смертні. Сократ – людина. Отже, Сократ смертний.

Коректність цих логічних міркувань базується не тільки на правдивості окремих речень, або їх частин та функціональних зв'язках між ними, а і у внутрішній будові самих речень, а також на розумінні виразів «кожен», «всі» і т.д.. Але такого типу міркування легко обґрунтовуються в теорії предикатів.

Означення 4.1. Нехай $D_1, D_2, \dots, D_n$ – непорожні множини. n-арним предикатом P^n, $n \geq 0$ називається відображення з декартового добутку $D_1 \times D_2 \times \dots \times D_n$ в множину $\{0; 1\}$ ($\{false, true\}$). $P^n: D_1 \times D_2 \times \dots \times D_n \to \{0: 1\}$ Множини $D_1, D_2, \dots, D_n$ називаються предметними областями

При $n = 0$ маємо логічну константу $P^0 = 0$, або $P^0 = 1$. При $n = 1$ маємо унарний предикат, $n = 2$ – бінарний, $n = 3$ – тернарний і т.д.

У випадку, коли $D_1 = D_2 = \dots = D_n = D$ предикат називається однорідним. Насправді, неоднорідний предикат можна звести до однорідного, якщо $D = D_1 \cup D_2 \cup \dots \cup D_n$ та шляхом деякого ускладнення відповідних формул (див. 4.1.2).

Надалі, якщо не сказано інше, предикати вважаємо однорідними заданими на предметній області D.

Приклад 4.1. Наведемо кілька прикладів предикатів:

1. $D = \mathbb{N}$, $P^1(x) -$ «число x є повним квадратом»
   ($|P(4)| = 1, |P(10)| = 0$);
2. $D = \mathbb{R}$, $Q^2(x, y) = |x| \leq y = \begin{cases} 0, & \text{коли } |x| > y; \\ 1, & \text{коли } |x| \leq y. \end{cases}$
   ($|Q(1,2)| = 1$, $|Q(2,-1)| = 0$);
3. $D = \{\text{люди}\}$, $R^3(x, y, z) =$ « z є спільним знайомим x та y».

Бачимо, що при підстановці фіксованих значень з предметної області на місце аргументів предикат перетворюється у висловлення. Також на кожному n-арному відношенню R $(n \geq 1)$, можна побудувати так званий n-арний характеристичний предикат.

Означення 4.2. Якщо відношення $R \subset D_1 \times D_2 \times \ldots \times D_n$, то його характеристичний предикат $P_R: D_1 \times D_2 \times \ldots \times D_n \to \{0:1\}$ визначається наступним чином: $|P_R(x_1 \ldots x_n)| = 1 \Leftrightarrow (x_1, x_2 \ldots x_n) \in R$.

Означення 4.3. n-арним, або n-місним функтором на декартовому добутку $D_1 \times D_2 \times \ldots \times D_n$ називається відображення

$$f^n: D_1 \times D_2 \times \ldots \times D_n \to D,\ n \geq 0\ (D_i \neq \emptyset,\ i = \overline{1,n},\ D \neq \emptyset).$$

У випадку, коли $D_1 = D_2 = \cdots = D_n = D$ функтор називається однорідним. При $n = 0$ матимемо предметну константу f^0 (фіксований елемент з D).

Приклад 4.2.

1. $D_1 = D_2 = D = \mathbb{N}$, $f^2(x, y) = \text{НСК}(x, y)$.
2. $D_1 = \{\text{чоловіки}\}$, $D = \{\text{жінки}\}$, $f^1(x) =$ «теща x».

Ключовим значенням в теорії предикатів є поняття квантора.

Означення 4.4. Нехай $P(\cdot)$ – предикат (необов'язково унарний), тоді вираз $\forall x P(x)$ буде правдивим тоді і тільки тоді, коли для всіх x з предметної області буде правдивим вираз $P(x)$. Вираз $\exists x P(x)$ правдивий тоді і тільки тоді, коли вираз $P(x)$ буде правдивим хоча б для одного x з предметної області.

Символ «∀» будемо називати *квантором загальності*, а «∃» – *квантором існування*. Між собою вони зв'язані такими співвідношеннями:

$$\forall x P(x) = \neg(\exists x \neg P(x)),\ \exists x P(x) = \neg(\forall x \neg P(x)).$$

4.1.2. Рекурсивне означення формули в алгебрі предикатів

Алфавіт алгебри предикатів містить наступні об'єкти:

1) предметні константи (маленькі літери на початку латинської абетки з індексами, або без індексів: a, b, c_1, c_2,…);
2) Предметні змінні (маленькі літери в кінці латинської абетки з індексами, або без індексів: x, y, z_1, z_2,…);
3) Функціональні символи (літери f, g, h з індексами, або без індексів, наприклад, f_1^2 – перший бінарний функціональний символ f);
4) Предикатні символи (великі латинські літери з індексами, або без індексів, наприклад, P_2^1 – другий унарний предикатний символ P);
5) Логічні зв'язки: $\vee, \wedge, \neg$, а також $\rightarrow, \leftrightarrow, \oplus$;
6) Квантори: $\forall, \exists$;

Допоміжні символи: «(», «)», «,».

Означення 4.5. (рекурсивне означення терму).

1. Предметна константа та предметна змінна – терм.
2. Якщо $t_1, t_2, \dots, t_n$ – терми, а f^n – n-арний функціональний символ, то $f^n(t_1, t_2, \dots, t_n)$ – терм.
3. Інших термів немає.

Терм – об'єкт, який можна підставляти на місце аргументу в формулі алгебри предикатів (див. Означення 4.6).

Означення 4.6. (рекурсивне означення формули).

1. Нехай P^n – n-арний предикатний символ, $t_1, t_2, \dots, t_n$ – терми, тоді $P^n(t_1, t_2, \dots, t_n)$ – формула, яку будемо називати атомарною, або атомом.

   Літерою будемо називати атом або його заперечення.
2. Якщо $\mathcal{A}, \mathcal{B}$ – формули, а x – предметна змінна, то формулами будуть $(\mathcal{A} \vee \mathcal{B})$, $(\mathcal{A} \wedge \mathcal{B})$, $(\neg\mathcal{A})$, $(\exists x\mathcal{A})$, $(\forall x\mathcal{A})$.
3. Інших формул немає.

Зауваження 4.1. Надалі домовимося зовнішні дужки опускати. Також вважаємо, що $\neg$ має пріорітет перед $\vee$ та $\wedge$ і для формули $\forall x\mathcal{A}$, або $\exists x\mathcal{A}$ всі входження змінної x в $\mathcal{A}$ попадають під дію квантора.

Означення 4.7. Змінна, яка попадає під дію хоча б одного квантора називається зв'язаною, в протилежному випадку – вільною.

Означення 4.8. Формула, яка не містить жодної вільної змінної (усі змінні зв'язані) називається замкнутою.

Приклад 4.3. Розглянемо формули:

1) $\forall x A(x, y)$;
2) $\forall x(\exists y A(x, y) \wedge B(x, y))$;
3) $\forall x \exists x(A(x) \vee B(x))$.

У першій формулі змінна x зв'язана, а y вільна У другій – обидва входження x зв'язані також зв'язаним є перше входження y (в $A(x,y)$), але друге входження y (в $B(x,y)$) – вільне. Формула 3) замкнута, бо обидва входження змінної x $(A(x), B(x))$ попадають під дію квантора існування, і тому $\forall x \exists x\big(A(x) \vee B(x)\big) = \exists x(A(x) \vee B(x))$.

Приклад 4.4. Твердження про те, що

$$\lim_{n\to\infty} a_n = a \Leftrightarrow \forall \varepsilon > 0\ \exists m \in \mathbb{N}: \forall n \in \mathbb{N}, n \geq m\ |a_n - a| < \varepsilon$$

В алгебрі предикатів виглядає наступним чином:

$$\forall \varepsilon((\varepsilon > 0) \to \exists m((m \in \mathbb{N}) \wedge \forall n(((n \in \mathbb{N}) \wedge (n \geq m)\ ) \to$$

$$\to (|\ a(n) - a| < \varepsilon)))\ ,$$

або, якщо позначити $G(x,y) =$ «$x < y$» $N(x) =$ «$x \in \mathbb{N}$», $f(x,y) = |x - y|$, матимемо

$$\forall \varepsilon(G(\varepsilon, 0) \to \exists m((N(m) \wedge \forall n((N(n) \wedge \neg G(m,n)) \to G(\varepsilon, f(a(n), a))))).$$

4.2. Інтерпретація формул алгебри предикатів

Означення 4.9. Задати інтерпретацію формули алгебри предикатів означає:

1. Задати предметну область, яку в подальшому будемо називати областю інтерпретації D.

2. Кожній константі даної формули співставляємо. фіксований елемент з області інтерпретації $a \mapsto |a| \in D$.
3. Кожному функціональному символу, який міститься в формулі ставимо у відповідність відображення (n-арний функтор) з $D^{\times n}$ в D, $f^n \mapsto |f^n|: D^{\times n} \to D$ $(n \geq 0)$.
4. Кожному n-арному предикатному символу ставимо у відповідність n-арний предикат на D, $P^n \longmapsto |P^n|: D^{\times n} \to \{0:1\}$ $(n \geq 0)$.
5. Змінні не інтерпретуються. Вони можуть набувати довільних значень на області інтерпретації.

Тобто кожній інтерпретації формула перетворюється в предикат, який залежить від вільних змінних вихідної формули. Зокрема, замкнута формула на кожній інтерпретації перетворюється на висловлення.

Крім інтерпретації (повної інтерпретації) розглядаються також часткову інтерпретацію формул алгебри предикатів, коли інтерпретуються не всі об'єкти.

Приклад 4.5. На області інтерпретації $D = \{a, b\}$ виписати всі інтерпретації формул $F_1 = \forall x P(x)$ та $F_2 = \exists x P(x)$. Дані формули містять лише один предикатний символ тому існує чотири інтерпретації $P \longmapsto |P|: \{a, b\} \to \{0:1\}$ (табл. 4.1).

Табл. 4.1. Інтерпретації формул F_1 та F_2

x	P_0	P_1	P_2	P_3
a	0	0	1	1
b	0	1	0	1
F_1	0	0	0	1
F_2	0	1	1	1

У два нижні рядки табл. 4.1 записані значення формул F_1 та F_2 на кожній інтерпретації.

Приклад 4.6. Виписати всі інтерпретації формули $F_1 = \forall x \exists y P(x, y)$ на області $\{a, b\}$. Дана формула містить один бінарний предикатний символ, тому існує 16 інтерпретацій $P \longmapsto |P|: \{a, b\}^2 \to \{0: 1\}$ (табл. 4.2).

Табл. 4.2. Інтерпретації формули F_1 з прикладу 4.6.

x	y	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
a	a	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
a	b	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
b	a	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
b	b	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
F_1		0	0	0	0	0	1	1	1	0	1	1	1	0	1	1	1

Пояснення. Розглянемо випадки I_1: P_8 та I_2: P_9. На I_1 формула хибна ($|F_1|_{I_1} = 0$), для $x = a$ можна знайти $y = a$, щоб була правдива $P(x, y)$, бо $|P(a, a)|_{I_1} = 1$. Але для $x = b$ ми жодного y не знайдемо ($|P(b, a)|_{I_1} = |P(b, b)|_{I_1} = 0$).

А у випадку I_2 для $x = a$ знаходиться $y = a$, та для $x = b$ знаходиться $y = b$, бо $|P(a, a)|_{I_2} = |P(b, b)|_{I_2} = 1$). Тому для кожного $x \in \{a, b\}$. можемо знайти y, щоб була правдивою формула $P(x, y)$. Тому F_1 на заданій інтерпретації правдива ($|F_1|_{I_2} = 1$).

Вправа 4.1. Виписати всі інтерпретації формули $F_2 = \exists y \forall x P(x, y)$ на області $\{a, b\}$.

Приклад 4.7. $F = \exists x \big(P(x, a) \wedge \forall y G(f(x), y)\big)$.

Визначити правдивість формули F на інтерпретаціях I_1 та I_2:

I_1: $D = \mathbb{N}$, $a = 5$, $P(x, y) =$ «$y \vdots x$», $f(x) = x^2$, $Q(x, y) =$ «$x \leq y$».

I_2: $D = \mathbb{R}$, $a = \pi$, $P(x, y) =$ «$x^2 + y^2 \leq 1$», $f(x) = sin\, x$,

$Q(x, y) =$ «$[x] = [y]$». Тут $[x]$ – ціла частина x.

Розглянемо I_1.

$|F|_{I_1}$=$\exists x((5 \vdots x) \wedge \forall y(x^2 \leq y))$.

$|F|_{I_1} = 1$, бо можна знайти $x = 1 \in \mathbb{N}$ таке, що $5 \vdots 1$ і для всіх $y \in \mathbb{N}$ виконується нерівність $1^2 \leq$ y.

Розглянемо I_2.

$|F|_{I_2}$=$\exists x((x^2 + \pi^2 \leq 1) \wedge \forall y([sin\, x] = [y]))$.

Ми не можемо знайти жодного $x \in \mathbb{R}$, для якого буде справедливою умова $x^2 + \pi^2 \leq 1$, а тому $|F|_{I_2}$=0.

Означення 4.10. Формула алгебри предикатів називається логічно загальнозначущою (ЛЗЗ), якщо вона правдива на всіх інтерпретаціях та при всіх наборах значень своїх вільних змінних.

Означення 4.11. Формула алгебри предикатів називається суперечливою, якщо вона хибна на всіх інтерпретаціях та при всіх наборах значень своїх вільних змінних.

Означення 4.12. Формула алгебри предикатів виконується, якщо вона правдива хоча б на одній інтерпретації та хоча б при одному наборі значень своїх вільних змінних.

Означення 4.13. Дві формули F_1 та F_2 називаються еквівалентними (логічно еквівалентними), якщо вони одночасно правдиві або хибні на кожній інтерпретації та при кожному наборі своїх вільних змінних (позначається $F_1 = F_2$, або $F_1 \Leftrightarrow F_2$).

Зрозуміло, що F – ЛЗЗ формула тоді і тільки тоді, коли $\neg F$ – суперечлива; F виконується тоді і тільки тоді, коли F не є суперечливою; $F_1 = F_2$ тоді і тільки тоді, коли $F_1 \leftrightarrow F_2$ – ЛЗЗ формула.

Далі випишемо деякі приклади ЛЗЗ формул та покажемо, як можна довести, або спростувати логічну загально значущість формул засобами алгебри предикатів.

4.3. Перевірка загальнозначущості формул алгебри предикатів

4.3.1. Основні ЛЗЗ формули алгебри предикатів

1.	$\forall x P(x) \to P(y)$	Правило універсальної конкретизації
2.	$P(a) \to \exists x P(x)$	Правило екзистенціального узагальнення
3.	а) $\neg\forall \times P(x) \leftrightarrow \exists x \neg P(x)$ б) $\neg\exists x P(x) \leftrightarrow \forall x \neg P(x)$	закон де Моргана
4.	а) $\forall x(P(x) \wedge Q(x)) \leftrightarrow (\forall x P(x) \wedge \forall x Q(x))$ б) $\exists x(P(x) \vee Q(x)) \leftrightarrow (\exists x P(x) \vee \exists x Q(x))$	
5.	а) $(\forall x P(x) \vee \forall x Q(x)) \to \forall x(P(x) \vee Q(x))$ б) $\exists x(P(x) \wedge Q(x)) \to (\exists x P(x) \wedge \exists x Q(x))$	
6.	а) $\forall x\big(P(x) \to Q(x)\big) \to \big(\forall x P(x) \to \forall x Q(x)\big)$	

б) $\left(\exists x P(x) \rightarrow \exists x Q(x)\right) \rightarrow \exists x\left(P(x) \rightarrow Q(x)\right)$

7. а) $\forall x\left(P(x) \leftrightarrow Q(x)\right) \rightarrow \left(\forall x P(x) \leftrightarrow \forall x Q(x)\right)$

б) $\left(\exists x P(x) \leftrightarrow \exists x Q(x)\right) \rightarrow \exists x\left(P(x) \leftrightarrow Q(x)\right)$

8. а) $\forall x(P(x) \wedge B) \leftrightarrow (\forall x P(x) \wedge B)$ Формула B не містить входжень x

б) $\forall x(P(x) \vee B) \leftrightarrow (\forall x P(x) \vee B)$

в) $\exists x(P(x) \wedge B) \leftrightarrow (\exists x P(x) \wedge B)$

г) $\exists x(P(x) \vee B) \leftrightarrow (\exists x P(x) \vee B)$

д) $\forall x(P(x) \rightarrow B) \leftrightarrow (\exists x P(x) \rightarrow B)$

е) $\exists x(P(x) \rightarrow B) \leftrightarrow (\forall x P(x) \rightarrow B)$

9. а) $\forall x \forall y P(x, y) \leftrightarrow \forall y \forall x P(x, y)$

б) $\exists x \exists y P(x, y) \leftrightarrow \exists y \exists x P(x, y)$

в) $\exists y \forall x P(x, y) \rightarrow \forall x \exists y P(x, y)$

10. а) $\forall x P(x) \leftrightarrow \forall y P(y)$ Жодне вільне входження y не стане зв'язаним в результаті заміни x на y

б) $\exists x P(x) \leftrightarrow \exists y P(y)$

4.3.2. Перевірка ЛЗЗ формул засобами алгебри предикатів

Суть цього методу полягає в тому, що ми ставимо під сумнів ЛЗЗ формули і намагаємося знайти таку інтерпретацію і такий набір значень вільних змінних, при яких формула хибна. Якщо нам це вдалося зробити, то формула не ЛЗЗ. А коли не вдалося, тобто прийшли до суперечності, то

наша формула є ЛЗЗ (правдива на всіх інтерпретаціях і при всіх наборах значень своїх вільних змінних).

Приклад 4.8. Доведемо ЛЗЗ формули 5б) і покажемо, що зворотна імплікація не є ЛЗЗ формулою. Позначимо через $G_1 = \exists x(P(x) \wedge Q(x))$, $G_2 = \exists x P(x) \wedge \exists x Q(x)$, $F_1 = G_1 \to G_2$, $F_2 = G_2 \to G_1$.Відмітимо, що F_1 і F_2 – замкнуті формули.

Доведемо ЛЗЗ формули F_1. Припустимо супротивне:

1.	$\lvert F_1\rvert = 0$	Припущення (Пр)
2.	$\lvert G_1\rvert = 1$	(1)
3.	$\lvert G_2\rvert = 0$	(1)
4.	$\lvert P(a) \wedge Q(a)\rvert = 1$	(2)
5.	$\lvert P(a)\rvert = 1$	(4)
6.	$\lvert Q(a)\rvert = 1$	(4)
7.	$\lvert \exists x P(x)\rvert = 1$	(5)
8.	$\lvert \exists x Q(x)\rvert = 1$	(6)
9.	$\lvert G_2\rvert = 1$	(7,8)

Пункти 3 та 9 дають суперечність, тобто припущення про те, що знайдеться хоча б одна інтерпретація, на якій F_1 хибна не вірне. А отже формула F_1 – правдива на всіх інтерпретаціях, тобто є ЛЗЗ.

Розглянемо F_2.

1.	$\lvert F_2\rvert = 0$	Пр
2.	$\lvert G_2\rvert = 1$	(1)
3.	$\lvert G_1\rvert = 0$	(1)
4.	$\lvert \exists x P(x)\rvert = 1$	(2)
5.	$\lvert \exists x Q(x)\rvert = 1$	(2)
6.	$\lvert P(a)\rvert = 1$	(4)

7. $|Q(b)| = 1$ (5)
8. $|P(a) \wedge Q(a)| = 0$ (3)
9. $|P(b) \wedge Q(b)| = 0$ (3)
10. $|Q(a)| = 0$ (6,8)
11. $|P(b)| = 0$ (7,9)

Формула $F_2 -$ не ЛЗЗ, бо знайшлася інтерпретація I, на якій вона хибна. Ця інтерпретація виглядає наступним чином: I: $D = \{a, b\}$,

x	P_2	Q_1
a	1	0
b	0	1

Вправа 4.2. Довести ЛЗЗ інших формул з 4.3.1.

4.4. Логічний наслідок в алгебрі предикатів

Означення 4.14. Формула B логічно випливає з формул $A_1, A_2, \ldots, A_n, n \geq 0$, якщо B правдива на всіх інтерпретаціях і при всіх значеннях вільних змінних, на яких всі формули $A_1, A_2, \ldots, A_n$ – правдиві одночасно. Сам логічний наслідок будемо позначати таким чином. $A_1, A_2, \ldots, A_n \vDash B$. Тут формули $A_1, A_2, \ldots A_n$ називаються гіпотезами а формула B – наслідком.

Випадок $n = 0$, тобто $\vDash B$ означає, що B – ЛЗЗ формула. При $n = 1$ будемо писати $A \Longrightarrow B$. Зрозуміло, що $(A \Longrightarrow B) \wedge (B \Longrightarrow A)$ тоді і тільки тоді, коли $A \Leftrightarrow B$ (формули A та B логічно еквівалентні).

Безпосередньо з означення логічного наслідку, чи логічної еквівалентності випливає теорема дедукції.

Теорема 4.1. (теорема дедукції для алгебри предикатів).

1. Має місце логічний наслідок $A_1, A_2, \ldots A_n \vDash B$ тоді і тільки тоді, коли формула $(A_1 \wedge A_2 \wedge \ldots \wedge A_n) \to B$ – ЛЗЗ.
2. $A \Leftrightarrow B$ тоді і тільки тоді, коли формула $A \leftrightarrow B$ – ЛЗЗ.

Означення 4.15. Говорять, що змінна y вільна для x. в формулі $\mathcal{A}(x)$, якщо при заміні x на y жодне вільне входженні x не зв'язується в результаті заміни.

Приклад 4.9. Розглянемо формулу $\exists x(2x = y)$, $x,y \in \mathbb{R}$.
У цій формулі z вільна для y: $\exists x(2x = z)$ але y не вільна для x, бо в результаті заміни x на y отримаємо зовсім іншу формулу $\exists y(2y = y)$.

Випишемо кілька основних правил виведення алгебри предикатів:

1.	$\forall x A(x) \Longrightarrow A(y)$	правило універсальної конкретизації
2.	$\exists x A(x) \Longrightarrow A(b)$, $b \in D$	правило екзистенціальної конкретизації
3.	$A(y) \Longrightarrow \exists x A(x)$, де x вільна для y в $A(y)$	правило екзистенціального узагальнення
4.	$C \to A(x) \Longrightarrow C \to \forall x A(x)$	правило загальності
5.	$A(x) \to C \Longrightarrow \exists x A(x) \to C$	правило існування

6. Якщо $\Gamma \vDash A(x)$, то $\Gamma \vDash \forall x A(x)$, коли x не вільна в будь-якій з формул Γ — правило узагальнення (Gen)

Доведемо, наприклад, 3.

Якщо на деякій інтерпретації $|A(y)| = 1$, $y \in D$, то існує $y = b$, $b \in D$ таке, що $|A(b)| = 1$. І тому $|\exists x A(x)| = 1$ на D.

Вправа 4.3. Довести інші логічні наслідки 1, 2, 4 – 6.

4.5. Формалізація речень природної мови. Задачі на логічний наслідок

Приклад 4.10. Розглянемо область визначення $D = \{$*люди*$\}$ та введене позначення: $J(x)$ – «x – суддя»; $L(x)$ – «x – юрист»; $S(x)$ – «x – злодій»; $A(x,y)$ – «x любить y». Запишемо наступні речення у вигляді формул:

1. Кожен суддя – юрист $\forall x\big(J(x) \to L(x)\big)$.
2. Деякі юристи – судді $\exists x\big(L(x) \wedge J(x)\big)$.
3. Деякі злодії люблять всіх юристів $\exists x\Big(S(x) \wedge \forall y\big(L(y) \to A(x,y)\big)\Big)$.
4. Деякі злодії люблять тільки юристів $\exists x\Big(S(x) \wedge \forall y\big(A(x,y) \to L(y)\big)\Big)$.
5. Деякі злодії люблять деяких юристів $\exists x\Big(S(x) \wedge \exists y\big(L(y) \wedge A(x,y)\big)\Big)$.
6. Всі злодії не люблять суддів $\forall x\Big(S(x) \to \forall y\big(J(y) \to \neg A(x,y)\big)\Big)$.

Приклад 4.11. Задача про перукаря.

В місті N перукарі стрижуть тих і тільки тих, хто не стрижеться сам. Довести, що в місті N немає жодного перукаря.

Розв'язання. Область визначення $D = \{\text{мешканці міста N}\}$.

Позначимо через $P(x)$ – «x – перукар»; та $S(x,y)$ – «x стриже y».

Розіб'ємо нашу посилку на дві:

1. Кожен перукар стриже тих, хто не стрижеться сам.

$$F_1 = \forall x \forall y \Big(\big(P(x) \wedge \neg S(y,y) \big) \to S(x,y) \Big).$$

2. Жоден перукар не стриже тих, хто стрижеться сам.

$$F_2 = \forall x \forall y \Big(\big(P(x) \wedge S(y,y) \big) \to \neg S(x,y) \Big).$$

Наслідок виглядатиме так: $B = \forall x \neg P(x)$.

Нам потрібно довести логічний наслідок $F_1, F_2 \vDash B$. Припустимо, що логічний наслідок невірний. Тобто існує інтерпретація при яких формули F_1 та F_2 правдиві, а формула B – хибна.

1.	$\lvert F_1 \rvert = 1$	Г1
2.	$\lvert F_2 \rvert = 1$	Г2
3.	$\lvert B \rvert = 0$	Пр
4.	$\lvert P(a) \rvert = 1$	(3)
5.	$\lvert \big(P(a) \wedge \neg S(a,a) \big) \to S(a,a) \rvert = 1$	(1)
6.	$\big\lvert \big(P(a) \wedge S(a,a) \big) \to \neg S(a,a) \big\rvert = 1$	(2)
7.	$\lvert \neg S(a,a) \to S(a,a) \rvert = \lvert S(a,a) \rvert = 1$	(4,5)
8.	$\lvert S(a,a) \to \neg S(a,a) \rvert = \lvert \neg S(a,a) \rvert = 1$	(4,6)

Пункти 7 та 8 дають суперечність, а тому не існує жодної інтерпретації, яка спростовує логічний наслідок. Логічний наслідок доведено.

Дана задача ілюструє парадокс Б. Рассела який в популярному вигляді формулюється таким чином. В місті N перукарі стрижуть тих і тільки тих, хто не стрижеться сам. Питання: чи стрижуть перукарі самі себе?

4.6. Поняття про формальну теорію *K*. Числення предикатів першого порядку

4.6.1. Означення числення предикатів

Формальна теорія K будується таким чином:

I. Алфавіт теорії:

1. Не порожня, не більш ніж зліченна множина предикатних символів (P_k^n – k-й n-арний предикатний символ $n, k \geq 1$);
2. Не більш ніж зліченна, можливо порожня, множина констант $(a_1, a_2, \dots)$;
3. Не більш ніж злічена, можливо порожня, множина функціональних символів (f_k^n – k-й n-арний функціональний символ $n, k \geq 1$);
4. Зліченна множина змінних ($x_1, x_2, \dots$);
5. Логічні зв'язки: →, ¬;
6. Квантор загальності ∀;
7. Допоміжні символи: «(», «)», «,».

II. Означення формули:

Терм визначається таким же чином, як і в алгебрі предикатів.

Формула визначається аналогічно алгебрі предикатів з використанням зв'язок → та ¬ та квантора загальності.

Для скорочення запису також використовуються позначення $A \vee B$, $A \wedge B$, $A \leftrightarrow B$ введені в розділі 3, а також $\exists x A = \neg(\forall x \neg A)$.

III. Система аксіом складається з п’яти нескінченних серій:

A1: $A \to (B \to A)$;

A2: $\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)$;

A3: $(\neg B \to \neg A) \to \big((\neg B \to A) \to B\big)$;

A4: $\forall x A(x) \to A(t)$

(тут формула $A(t)$ отримана з $A(x)$ заміною всіх входжень змінної x на терм t, причому жодна змінна, яка входить в t не потрапляє під дію квантора формули A. Тобто змінні, які входять в t не повинні бути зв’язаними);

A5: $\forall x(A \to B) \to (A \to \forall x B)$

(тут формула A не містить вільних входжень змінної x).

A1 – A5 називають логічними аксіомами. Крім того, в кожній конкретній теорії K можуть бути свої власні аксіоми (приклади див. 4.6.5.).

Якщо ж власних аксіом немає, то маємо справу з теорією K, яка називається численням предикатів.

IV. Правила виведення:

$A, A \to B \vdash B$ MP (Modus Ponens);

$A \vdash \forall x A$ Gen (узагальнення).

(взагалі, в алгебрі предикатів $A \nRightarrow \forall x A$).

4.6.2.Метатеорема дедукції для теорії K

Означення 4.16. Нехай A та B – формули теорії K, Γ – множини гіпотез (також формули з K) і має місце формальне виведення $\Gamma, A \vdash B$. Кажуть, що формула B залежать від A якщо:

1. B виводиться з A за допомогою одного з правил виводу;
2. Формула B виводиться з формул залежних від A за допомогою одного з правил виведення.

У формальній теорії K також справедлива метатеорема дедукції, але у слабкішому вигляді, ніж в теорії L.

Теорема 4.2. (МТД для формальної теорії K).

Нехай в теорії K має місце формальне виведення $\Gamma, A \vdash B$ і в цьому виводі при застосуванні правила Gen до формул залежних від A не зв'язуються вільні змінні самої формули A, тоді має місце формальне виведення $\Gamma \vdash A \to B$.

Зокрема, твердження теореми 4.2. справедливе, коли правило Gen до формул залежних від A та до самої A не застосовується.

Приклад 4.12. З того, що $A(x) \vdash \forall x A(x)$ не випливає $\vdash A(x) \to \forall x A(x)$, але із $A(x) \vdash \forall y A(x)$ випливає $\vdash A(x) \to \forall y A(x)$.

4.6.3.Формальне виведення в теорії K

Доведемо спочатку правило універсальної конкретизації (УК) $\forall x\ A(x) \vdash A(y)$.

1. . $\forall x A(x)$ Г1

2. . $\forall x A(x) \to A(y)$ Г2

3. $A(y)$ МР (1,2)

Обґрунтування правила екзистенціальної конкретизації (ЕК) $\exists x\, A(x) \vdash A(b)$, $b \in D$ можна знайти в [1, с. 83-85].

Побудуємо формальне виведення для наступної задачі.

Приклад 4.13. Деякі студенти люблять своїх викладачів. Жоден студент не любить невігласів. Отже, жоден викладач не є невігласом.

Формалізуємо: $D = \{\text{люди}\}$, $S(x)$ – «x – студент», $T(x)$ – «x – викладач», $Q(x)$ – «x – невіглас», $L(x, y)$ – «x любить y».

гіпотези: F_1: $\exists x \left(S(x) \wedge \forall y\big(T(y) \to L(x, y)\big)\right)$;

F_2: $\forall x \left(S(x) \to \forall y\big(Q(y) \to \neg L(x, y)\big)\right)$.

Наслідок: G: $\forall y \left(T(y) \to \neg Q(y)\right)$.

Формальне виведення:

1. . $\exists x \left(S(x) \wedge \forall y\big(T(y) \to L(x, y)\big)\right)$ Г1

2. . $\forall x \left(S(x) \to \forall y\big(Q(y) \to \neg L(x, y)\big)\right)$ Г2

3. $S(b) \wedge \forall y\big(T(y) \to L(b, y)\big)$ ЕК(1)

4. . $S(b)$ $\wedge^{-}$(3)

5. . $\forall y\big(T(y) \to L(b, y)\big)$ $\wedge^{-}$(3)

6. . $S(b) \to \forall y\big(Q(y) \to \neg L(b, y)\big)$ УК(2)

7. . $\forall y\big(Q(y) \to \neg L(b, y)\big)$ МР(4,6)

8. .	$Q(z) \to \neg L(b,z)$	УК(7)
9.	$T(z) \to L(b,z)$	УК(5)
10.	$L(b,z) \to \neg Q(z)$	Контрапозиція (8)
11.	$T(z) \to \neg Q(z)$	В1 (9,10)
12.	$\forall y\big(D(y) \to \neg Q(y)\big)$	Gen (11)

Вправа 4.4. Твердження про те, що $A \to \neg B \vdash B \to \neg A$ (контрапозиція) пропонується довести самостійно.

4.6.4. Властивості числення предикатів

1. Числення предикатів несуперечливе ($\nexists A(\vdash A) \wedge (\vdash \neg A)$).
2. Числення предикатів неповне в широкому сенсі ($\neg\big(\forall A\ (\vdash A) \vee (\vdash \neg A)\big)$).
3. **Теорема 4.3.** (теорема Геделя[4] про повноту числення предикатів) Числення предикатів повне відносно алгебри предикатів, тобто формула є теоремою в численні предикатів тоді і тільки тоді, коли вона є ЛЗЗ в алгебрі предикатів.
4. Числення предикатів нерозв'язне, бо немає ефективної процедури, яка дозволяє визначити чи є формула теоремою, але є напіврозв'язним. Це означає, що існує ефективна процедура, яка підтверджує, що дана

[4] Курт Гедель (1906 – 1978) – австрійський логік, математик та філософ математики. Найбільш відомий сформульованою та доведеною теоремою про неповноту формальної арифметики.

формула є теоремою. Але в протилежному випадку ця процедура, взагалі кажучи, не зупиняється за скінченну кількість кроків.

4.6.5. Приклади формальних теорій *K*

1. Числення предикатів першого порядку. Власних аксіом немає.
2. Теорія з рівністю.

   K – теорія першого порядку. Є єдиний предикатний символ A_1^2. Для скорочення запису замість $A_1^2(t,s)$ пишемо $t = s$ і замість $\neg A_1^2(t,s)$ – $t \neq s$, констант і функціональних символів немає. До списку аксіом A1 – A5 додаються ще дві:

   A6: $\forall x_1(x_1 = x_1)$ (рефлексивність рівності)

   A7: $(x = y) \to \big(A(x,x) \to A(x,y)\big)$ (підстановність рівності),

   де x та y – предметні змінні, $A(x,x)$ – довільна формула, а $A(x,y)$ отримується з $A(x,x)$ заміною деяких (не обов'язково всіх) вільних входжень x входженнями y, за умови, що y вільний для тих входження x, які заміняються.
3. Теорія строгого порядку.

   Нехай K містить єдиний предикатний символ A_1^2 і не містить констант та функціональних символів. Замість $A_1^2(x_1,x_2)$ та $\neg A_1^2(x_1,x_2)$ будемо писати $x_1 < x_2$ та $x_1 \not< x_2$ відповідно. І ця теорія ще містить дві власні аксіоми:

A6: $\forall x_1(x_1 \not< x_1)$ (антирефлексивість);

A7: $\forall x_1 \forall x_2 \forall x_3\big((x_1 < x_2) \wedge (x_2 < x_3) \rightarrow (x_1 < x_3)\big)$ (транзитивність).

Будь-яка модель цієї теорії є впорядкованою структурою.

4. Формальна теорія груп.

Нехай теорія K має один предикатний символ A_1^2, один функціональний символ f_1^2 та одну предметну константу a_1. Будемо використовувати стандартні позначення $t = s$ замість $A_1^2(t, s)$, $t + s$ замість $f_1^2(t, S)$ та 0 замість a_1. Власні аксіоми:

A6: $\forall x_1 \forall x_2 \forall x_3 \Big(\big(x_1 + (x_2 + x_3)\big) = \big((x_1 + x_2) + x_3\big)\Big)$ (асоціативність);

A7: $\forall x_1(0 + x_1 = x_1)$;

A8: $\forall x_1 \exists x_2(x_1 + x_2 = 0)$ (існування оберненого);

A9: $\forall x_1(x_1 = x_1)$ (рефлексивність рівності);

A10: $\forall x_1 \forall x_2\big((x_1 = x_2) \rightarrow (x_2 = x_1)\big)$ (симетричність рівності);

A11: $\forall\, x_1 \forall x_2 \forall x_3 \Big((x_1 = x_2) \rightarrow \big((x_2 = x_3) \rightarrow (x_1 = x_3)\big)\Big)$

(транзитивність рівності);

A12: $\forall\, x_1 \forall x_2 \forall x_3((x_2 = x_3) \rightarrow ((x_1 + x_2 = x_1 + x_3) \wedge$

$\wedge(x_2 + x_1 = x_3 + x_1)))$

(підстановність рівності).

Будь яка модель даної теорії називається групою. Докладніше про приклади теорії K та їх властивості можна дізнатися, наприклад, в [1].

Розділ 5. Автоматичне доведення теорем

Як було вказано в попередньому розділі (пункт 4.6.4), числення предикатів є напіврозв'язним. Тобто існують ефективні процедури, які дозволяють підтверджувати, що формула є теоремою, що еквівалентно логічній загально значимості в алгебрі предикатів.

У цьому розділі мова буде вестися саме про такі алгоритми, але відносно суперечливості формул. Тому при доведенні ЛЗЗ формул слід не забувати брати заперечення.

Ці алгоритми можна використовувати і для ЕОМ як формалізацію логічного мислення. Найбільш повно дана тема висвітлюється в [3].

5.1. Попередні нормальні форми

Означення 5.1. Формула F в алгебрі предикатів записана у вигляді попередньої нормальної форми (ПНФ), якщо вона має вигляд $F = Q_1 x_1 Q_2 x_2 \cdots Q_n x_n M$, де Q_i – квантори ($\forall$, або $\exists$), x_i, $i = \overline{1,n}$ – предметні змінні, які попарно відрізняються, тобто $x_i \neq x_j$, при $i \neq j$, а також формула M не містить кванторів. Сама формула F має бути замкнутою (не містити вільних змінних). $Q_1 x_1 Q_2 x_2 \cdots Q_n x_n$ називають префіксом, а M – матрицею F.

Для зведення формули F до ПНФ потрібно:

1. Зв'язати вільні змінні F;
2. Перейменувати змінні, які відносяться до різних кванторів;

3. Винести всі квантори вперед.

Для реалізації пункту 1 допоможе наступна лема:

Лема 5.1. Формули $A(x)$ та $\exists x A(x)$ суперечливі або несуперечливі одночасно.

Доведення. Твердження леми 5.1. Випливає з означення квантора існування.

Нехай формула $\exists x A(x)$ суперечлива. Зафіксуємо інтерпретацію I та запишемо $|\exists x A(x)| = 0$. Це означає, що формула $|A(x)| = 0$ для всіх наборів значень своїх вільних змінних, включаючи x. А це означає, внаслідок вибору довільної інтерпретації I, суперечливість $A(x)$.

Якщо ж формула $\exists x A(x)$ несуперечлива, то існує деяка інтерпретація I, на якій $|\exists x A(x)| = 1$. Ця умова еквівалентна тому, що знайдеться значення змінної x з предметної області, коли $|A(x)| = 1$. А це означає несуперечливість формули $A(x)$.

Лема доведена.

Отже для реалізації пункту 1 слід для кожної вільної змінної поставити на початку формули квантор існування.

Для того, щоб безболісно можна було виносити квантори, слід перейменувати змінні, які відносяться до різних кванторів. А потім, щоб винести квантори вперед, слід використати правило де Моргана: (формули 3а), б) п. 4.3.1) та правила 8 а) – г) п. 4.3.1:

$$\neg \forall x P(x) \Leftrightarrow \exists x \neg P(x),\ \neg \exists x P(x) \Leftrightarrow \forall x \neg P(x)$$

$$\forall x P(x) \wedge B \Leftrightarrow \forall x (P(x) \wedge B),\ \ \forall x P(x) \vee B \Leftrightarrow \forall x (P(x) \vee B),$$

$$\exists x P(x) \wedge B \Leftrightarrow \exists x (P(x) \wedge B),\ \exists x P(x) \vee B \Leftrightarrow \exists x (P(x) \vee B).$$

Тут формула B не містить вільних входжень x, що забезпечується перейменуванням змінних.

Приклад 5.1. Звести до ПНФ формулу $F = \forall x P(x,y) \wedge \forall x Q(x)$.

Бачимо, що F містить вільну змінну y, тому зв'язуємо її квантором $\exists$.

$$F \rightsquigarrow \exists y\big(\forall x P(x,y) \wedge \forall x Q(x)\big).$$

Далі бачимо, що є дві змінні x, які попадають під дію різних кванторів, тому другий x замінимо на z і отримуємо $\exists y\big(\forall x P(x,y) \wedge \forall z Q(z)\big)$. Далі, користуючись правилом 8, виносимо квантори вперед:

$$\exists y\big(\forall x P(x,y) \wedge \forall z Q(z)\big) \Leftrightarrow \exists y \forall x\big(P(x,y) \wedge \forall z Q(z)\big) \Leftrightarrow$$
$$\Leftrightarrow \exists y \forall x \forall z\big(P(x,y) \wedge Q(z)\big).$$

Отриманий вираз – ПНФ.

Зауваження 5.1. Знаком $\rightsquigarrow$ будемо позначати нееквівалентні переходи, які зберігають суперечливість формул.

Приклад 5.2. Звести до ПНФ формулу $F = \exists x P(x) \to \exists x Q(x)$.

Дана формула не містить вільних змінних, а тому одразу переходимо до пункту 2. Розписуємо імплікацію та міняємо другий xна y.

$$F \Leftrightarrow \neg \exists x P(x) \vee \exists y Q(y) \Leftrightarrow \forall x \neg P(x) \vee \exists y Q(y)$$

(використали правило де Моргана). Далі, враховуючи комутативність операції $\vee$ маємо два шляхи:

а) $\forall x \neg P(x) \vee \exists y Q(y) \Leftrightarrow \forall x\big(\neg P(x) \vee \exists y Q(y)\big) \Leftrightarrow$
$\Leftrightarrow \forall x \exists y\big(\neg P(x) \vee Q(y)\big)$;

б) $\forall x \neg P(x) \vee \exists y Q(y) \Leftrightarrow \exists y\big(\forall x \neg P(x) \vee Q(y)\big) \Leftrightarrow$
$\Leftrightarrow \exists y \forall x\big(\neg P(x) \vee Q(y)\big)$.

В обох випадках отримали ПНФ. Різницю між випадками а) та б) буде видно в наступному параграфі.

5.2. Скулемівські стандартні форми

Означення 5.2. Формула F записана у вигляді скулемівської стандартної форми, якщо вона знаходиться в ПНФ та має вигляд:

$$F = \forall x_1 \forall x_2 \dots \forall x_n M,$$

де матриця M записана у вигляді КНФ (кон'юктивної нормальної форми).

Для зведення формули до ССФ потрібно:

1. Звести формулу до ПНФ (параграф 5.1);
2. Звести M до КНФ;
3. Позбутися кванторів існування (процедура елімінування кванторів існування).

Будемо вважати, що формула F знаходиться в ПНФ.

Для реалізації процедури елімінування кванторів існування нам знадобляться дві наступні леми:

Лема 5.2. Формули $\exists x A(x)$ та $A(c)$ суперечливі (несуперечливі) одночасно. Тут формула $A(c)$ отримана з шляхом заміни всіх вільних входжень змінної x на константуc, причому c не міститься в формулі $A(x)$.

Лема 5.3. Формули $\forall x_1 \forall x_2 \dots \forall x_n \exists x A(x_1, x_2, \dots, x_n, x) = F_1$ та $\forall x_1 \forall x_2 \dots \forall x_n\, A\left(\left(x_1, x_2, \dots, x_n, f(x_1, x_2, \dots, x_n)\right)\right) = F_2$ суперечливі

(несуперечливі) одночасно. Тут формула F_2 отримана з F_1 шляхом заміни всіх вільних входжень змінної x на функціональний символ $f(x_1 \dots x_n)$, який не міститься в формулі F_1.

Оскільки лема 5.2 є частинним випадком леми 5.3, доведемо лему 5.3.

Доведення леми 5.3.

1) Нехай формула F_1 суперечлива і нам потрібно довести суперечливість формули F_2. Зафіксуємо довільну інтерпретацію та набір значень вільних змінних (в формулі A можуть бути і інші змінні крім $x_1, x_2 \dots x_n$ та x).

Оскільки F_1 суперечлива, тоді

$$|F_1| = |\forall x_1 \forall x_2 \dots \forall x_n \exists x A(x_1, x_2, \dots, x_n, x)| = 0 \Leftrightarrow$$
$$\Leftrightarrow |\exists x_1 \exists x_2 \dots \exists x_n \forall x \neg A(x_1, x_2, \dots, x_n, x)| = 1.$$

Тобто знайдуться такі значення змінних $x_1, x_2, \dots, x_n \in D$ і для всіх значень $x \in D$ $|A(x_1, x_2, \dots, x_n, x)| = 0$, зокрема, і для $x = f(x_1, x_2, \dots, x_n)$. Тобто $\left|A\left(x_1, x_2, \dots, x_n, f(x_1, x_2, \dots, x_n)\right)\right| = 0$ при деяких фіксованих $x_1, x_2 \dots x_n \in D$, а тому $|\neg F_2| = 1 \Leftrightarrow |F_2| = 0$ на довільній інтерпретації і при довільних значення вільних змінних. Робимо висновок, що формула F_2 також суперечлива.

2) Нехай F_1 виконується. Потрібно довести, що F_2 також виконується.

Нехай існує деяка інтерпретація та набір значень вільних змінних, на якій $|F_1| = |\forall x_1 \forall x_2 \dots \forall x_n \exists x A(x_1, x_2, \dots, x_n, x)| = 1$.

Це означає, що для всіх значень змінних $x_1, x_2 \dots x_n \in D$ знайдеться хоча б одне значення $x \in D$ таке, що $|A(x_1, x_2, \dots, x_n, x)| = 1$.

Тобто ми можемо кожному набору $(x_1, x_2, \dots, x_n)$ співставити одне довільне значення x, яке позначимо $f(x_1, x_2, \dots, x_n)$ $((x_1, x_2, \dots, x_n) \longmapsto x =$

$= f(x_1, x_2, \dots, x_n)$), так, щоб $\left|A\left(x_1, x_2, \dots, x_n, f(x_1, x_2, \dots, x_n)\right)\right| = 1$ для всіх $x_1, x_2 \dots x_n \in D$. Тобто F_2. виконується.

Зауважимо, що функтор f можна будувати не єдиним чином, якщо деяким наборам $(x_1, \dots, x_n)$ можна підібрати декілька x таких, що $|A(x_1, x_2, \dots, x_n, x)| = 1$.

Лема доведена.

Як бачимо, будь-яка формула алгебри предикатів може зводитися до ССФ із збереженням суперечливості (несуперечливості).

Приклад 5.3. Звести формулу $F = \exists x P(x) \to \exists x Q(x)$ до ССФ (продовження прикладу 5.2).

Для даної форми ми отримали два варіанта ПНФ. В обох випадках зведемо до ССФ, використовуючи леми 5.2 та 5.3.

а) $\forall x \exists y\left(\neg P(x) \vee Q(y)\right) \rightsquigarrow \forall x\left(\neg P(x) \vee Q\left(f(x)\right)\right.$ – ССФ;

б) $\exists y \forall x\left(\neg P(x) \vee Q(y)\right) \rightsquigarrow \forall x\left(\neg P(x) \vee Q(a)\right)$ – ССФ.

Приклад 5.4. Звести до ССФ формулу

$$F = \exists x \forall y \exists z \exists u \forall v \exists w P(x, y, z, u, v, w).$$

Змінну x замінимо на константу a, z – на функціональний символ $f(y)$, u – на $g(y)$, w – на $h(y, v)$ та отримаємо ССФ.

$$\forall y \forall v P\left(a, y, f(y), g(y), v, h(y, v)\right).$$

Як бачимо, якщо лівіше від квантора існування немає квантора загальності, то відповідна змінна замінюється на константу, в протилежному випадку – на функціональні символи від всіх змінних з квантором загальності, які в ПНФ стоять лівіше заданої.

Приклад 5.5. Звести до ССФ формулу

$$F = \forall y \exists x P\big(x, f(y)\big) \vee \exists x P(a, x).$$

$$F \Leftrightarrow \forall y \exists x P\big(x, f(y)\big) \vee \exists z P(a, z) \Leftrightarrow$$

$$\Leftrightarrow \exists z \Big(\forall y \exists x P\big(x, f(y)\big) \vee P(a, z)\Big) \Leftrightarrow \exists z \forall y \exists x \Big(P\big(x, f(y)\big) \vee P(a, z)\Big) \Leftrightarrow$$

$$\Leftrightarrow \forall y \Big(P\big(g(y), f(y)\big) \vee P(a, b)\Big)$$

Тут послідовно формулу F звели до ПНФ і позбулися кванторів існування. Оскільки константа a та функціональний символ f вже присутні в F, то вводимо іншу константу b та функціональний символ g.

Вправа 5.1. Звести до ССФ формули:

$$F_1 = \big(\forall x A(x) \to \exists x B(x, a)\big) \to \forall x A\big(f(x)\big);$$

$$F_2 = \forall x A(x, a) \oplus \forall x B(x, y);$$

$$F_3 = \forall x P(x) \leftrightarrow \exists x Q\big(f(x, y)\big).$$

Зауваження 5.2. Константи та функціональні символи, введені в лемах 5.2 та 5.3 називаються скулемівськими.

Отже, ми навчилися зводити формули в алгебрі предикатів до ССФ.

Тобто $F \rightsquigarrow \forall x_1 \forall x_2 \dots \forall x_n M \Leftrightarrow \forall x_1 \forall x_2 \dots \forall x_n (D_1 \wedge D_2 \wedge \dots \wedge D_m)$, де D_i – диз'юнкти $i = \overline{1, m}$, $m \geq 0$, з яких складаються КНФ формули F.

Далі будемо працювати з множиною диз'юнктів $S = \{D_1, D_2, \dots, D_m\}$. Нашою метою буде навчитися перевіряти суперечливість множини S.

З означення логічних операцій $\vee$, $\wedge$, та квантора загальності слідує означення суперечливості множини диз'юнктів.

Означення 5.3. Множина диз'юнктів S називається суперечливою, якщо на кожній інтерпретації хоча б один диз'юнкт приймає значення нуль принаймні на одному наборі значень змінних, що входять в диз'юнкти з S.

Означення 5.4. Диз'юнкт, який тотожно дорівнює нулю, називатимемо порожнім та позначатимемо □.

Очевидно, що множина диз'юнктів, яка містить порожній диз'юнкт, завжди є суперечливою. В зворотній бік твердження невірне.

Контрприклад. $S = \{P, Q, \neg P \vee \neg Q\}$. При $|P| = 0$ спростовується перший диз'юнкт, при $|Q| = 0$ – другий, а при $|P| = |Q| = 1$ – третій.

Зауваження 5.3. Змінні в рамках одного диз'юнкту можна перейменовувати так, щоб різним змінним співставлялися різні нові назви. Тобто з диз'юнкта $P(x) \vee Q(x, y)$ можна отримати диз'юнкти $P(y) \vee Q(y, x)$, $P(x_1) \vee Q(x_1, y_1)$, але неможна отримати $P(z) \vee Q(z, z)$. При цьому множина S і нова множина S', отримана в результаті таких замін, будуть суперечливі (несуперечливі) одночасно.

Нагадаємо, що атомом (атомарною формулою) називається формула вигляду $P^n(t_1, t_2, \dots, t_n)$, де P^n – n-арний предикатний символ, $t_1, t_2, \dots, t_n$ – терми. Літерою будемо називати атом або його заперечення (див. означення 4.5 та 4.6).

5.3. *H*-інтерпретації

За означенням 5.3 множина диз'юнктів суперечлива тоді і тільки тоді, коли на кожній інтерпретації спростовується хоча б один диз'юнкт, а отже і на всіх предметних областях. Але неможливо розглядати всі інтерпретації на всіх областях. На щастя, є така спеціальна область, що умова суперечливості S еквівалентна тому, що S суперечлива на всіх

інтерпретаціях на цій області. Цю область називають ербранівським універсумом, а інтерпретації на ній – H-інтерпретаціями.

5.3.1. Ербранівський універсум

Означення 5.5. Нехай S – множина диз'юнктів і нехай H_0 – множина констант з S. Тоді для $n \in \mathbb{N}$

$$H_n = H_{n-1} \cup$$

$$\cup \left\{ f^m(t_1, t_2, \dots, t_m) \middle| \begin{array}{l} t_i \in H_{n-1}, i = \overline{1,m}, f^m - m\text{-арний функціональний символ з } S, m \geq 0 \end{array} \right\}.$$

Якщо S не містить жодної константи, то вводимо самі одну константу, наприклад, a, і $H_0 = \{a\}$. Множина H_n називається множиною констант n-го рівня для S. І, нарешті, множину $H_\infty = \bigcup_{n=0}^{\infty} H_n$ називатимемо ербранівським універсумом для S.

Приклад 5.6. Нехай $S = \{P(x), \neg P(a) \vee P(f(x))\}$.

Тоді $H_0 = \{a\}$, $H_1 = \{a, f(a)\}$, $H_2 = \{a, f(a), f(f(a))\}$, … , $H_\infty = \{a, f(a), f(f(a)), \cdots\}$.

Приклад 5.7. Нехай $S = \{P(x) \vee Q(y),\ \neg Q(z) \vee R(u), \neg P(v) \vee \neg R(w)\}$ Тоді $H_0 = H_1 = H_2 = \cdots = H_\infty = \{a\}$.

Приклад 5.8. Нехай $S = \{P(g(x,a)), \neg P(f(b))\}$.

Тоді $H_0 = \{a, b\}$, $H_1 = \{a, b, f(a), f(b), g(a,a), g(a,b), g(b,a), g(b,b)\}$,
$H_2 = \{ a, b, f(a), f(b), g(a,a), g(a,b), g(b,a), g(b,b), f(f(a)),$
$f(f(b)), f(g(a,a)), f(g(a,b)), f(g(b,a)), f(g(b,b)), g(a, f(a)), \dots,$
$g(a, g(b,b)), g(b, f(a)), \dots, g(b, g(b,b)), (f(a), a), \dots, g(g(b,b), a), g(f(a), b), \dots,$

$g\ g(g(b,b),b), g\big(f(a),f(a)\big), \dots, g\big(g(b,b),g(b,b)\big)\}$ – всього 74 елемента. H_3 буде містити 2+74+74^2=5552 елемента.

. . .

Вправа 5.2. Довести, що множина H_∞ скінченна, або злічена.

5.3.2. Ербранівський базис

Означення 5.6. Нехай S – множина диз'юнктів. Тоді множина

$$\mathcal{A} = \left\{ \begin{array}{c} P^n(t_1, t_1, \dots, t_n) | \\ t_i \in H_\infty, i = \overline{1,n}, P^n - n\text{–арний } \textit{предикатний символ в } S, n \geq 0 \end{array} \right\}$$

називається ербранівським базисом.

Зауваження 5.4. При виписуванні ербранівського базису важливо слідкувати, щоб кожен його елемент зустрічається на скінченному кроці.

Приклад 5.9. Для $S = \{P(x), \neg P(a) \vee P\big(f(x)\big)\}$

$$H_\infty = \{a, f(a), f\big(f(a)\big), \cdots\},$$

$$\mathcal{A} = \{P(a), P\big(f(a)\big), P\left(f\big(f(a)\big)\right), P\left(f\left(f\big(f(a)\big)\right)\right), \dots\}.$$

Приклад 5.10. Для $S = \{P(x) \vee Q(y), \neg Q(z) \vee R(u), \neg P(v) \vee \neg R(w)\}$

$$H_\infty = \{a\}, \quad \mathcal{A} = \{P(a), Q(a), R(a)\}.$$

Приклад 5.11. Для $S = \{P\big(g(x,a)\big), \neg P(f(b))\}$.

$$\mathcal{A} = \left\{ \begin{array}{c} P(a), P(b), P\big(f(a)\big), P\big(f(b)\big), P\big(g(a,a)\big), \\ P\big(g(a,b)\big), P\big(g(b,a)\big), P\big(g(b,b)\big), \dots \end{array} \right\}.$$

5.3.3. Поняття про H-інтерпретації

Означення 5.7. Нехай S – множина диз'юнктів. H-інтерпретація на S задається наступним чином:

Предметна область – ербранівський універсум, $D = H_\infty$.

Всі константи з H_0 переходять в себе $a \mapsto a \in H_\infty$.

Нехай f^m – m–арний функціональний символ і $t_1, t_2, \dots, t_m \in H_\infty$, $m \in \mathbb{N}$, тоді f^m ставиться у відповідність функтор, який відображає довільний набір $(t_1, t_2, \dots, t_m)$ в $f(t_1, t_2, \dots, t_m)$.

І лише немає обмежень на інтерпретацію предикатних символів. Кожному n–арному предикатному символу ставиться у відповідність n–арний предикат $P^n \mapsto |P^n| : H_\infty^{\times n} \to \{0,1\}$, $n \geq 0$.

Для виписування різноманітних H-інтерпретацій зручно використовувати ербранівський базис.

Нехай, $\mathcal{A} = \{A_1, A_2, \dots, A_n, \dots\}$ – ербранівський базис множини S. Тоді I_H зручно записувати у вигляді $I_H = \{\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n \dots\}$, де $\tilde{A}_i = \begin{cases} A_i, & якщо\ |A_i|_{I_n} = 1 \\ \neg A_i, & якщо\ |A_i|_{I_n} = 0 \end{cases}$, $i \in \mathbb{N}$.

Насправді у випадках нескінченого ербранівського базису ми будемо працювати з частковими H-інтерпретаціями і співставляється значення 0 або 1 лише деяким елементам з $\mathcal{A}$.

Приклад 5.12. *Для* $S = \{P(x), \neg P(a) \vee P(f(x))\}$.

$\mathcal{A} = \{P(a), P(f(a)), P(f(f(a))), P(f(f(f(a)))), \dots\}$ – ербранівський базис.

Приклади інтерпретацій:

$$I_{H_1} = \left\{P(a), P\big(f(a)\big), P\left(f\big(f(a)\big)\right), P\left(f\left(f\big(f(a)\big)\right)\right), \dots\right\};$$

$$I_{H_2} = \left\{\neg P(a), \neg P\big(f(a)\big), \neg P\left(f\big(f(a)\big)\right), \neg P\left(f\left(f\big(f(a)\big)\right)\right), \dots\right\};$$

$$I_{H_3} = \left\{P(a), \neg P\big(f(a)\big), P\left(f\big(f(a)\big)\right), \neg P\left(f\left(f\big(f(a)\big)\right)\right), \dots\right\}.$$

. . .

Інтерпретації для S не обов'язково задаються на H_∞, тобто вони не є H-інтерпретаціями, але для кожної такої інтерпретації I можна побудувати відповідну H-інтерпретацію I_H.

Означення 5.8. Нехай I – інтерпретація для множини диз'юнктів S на предметній області D. H-інтерпретацією I_H, яка відповідає I, є інтерпретація, що задовольняє наступні умови:

Нехай $t_1, t_2, \dots, t_n \in H_\infty$, і нехай t_i відображається в деякий елемент $d_i \in D$ для всіх $i = \overline{1,n}$. $|P^n(d_1, d_2, \dots, d_n)|_I = 0(1)$ тоді і тільки тоді, коли $|P^n(t_1, t_2, \dots, t_n)|_{I_H} = 0(1)$. Тобто предикатні символи на відповідних наборах інтерпретуються однаково.

Лема 5.4. Якщо множина диз'юнктів S для деякої інтерпретації I на області D виконується, то S виконується і на довільній H-інтерпретації, яка відповідає I.

Вправа 5.3. Довести лему 5.4. самостійно.

Проілюструємо приклад побудови відповідної H-інтерпретації на прикладі.

Приклад 5.13. Нехай

$$F = \forall x \left(P(x) \oplus P\big(f(x)\big)\right).$$

Неважко пересвідчитися, що F виконується на такій інтерпретації I:

$D = \{a, b\}$

x	P_1	f_2
a	0	b
b	1	a

Побудуємо I_H – H-інтерпретацію, яка відповідає I. Зведемо формули до ССФ, використовуючи формулу $A \oplus B = (A \vee B) \wedge (\neg A \vee \neg B)$.

$$F \Leftrightarrow \forall x \Big(\big(P(x) \vee P\big(f(x)\big) \big) \wedge \big(\neg P(x) \vee \neg P\big(f(x)\big) \big) \Big).$$

$S = \{ P(x) \vee P\big(f(x)\big), \neg P(x) \vee \neg P\big(f(x)\big) \}$ – множина диз'юнктів,

$H_\infty = \{ c, f(c), f\big(f(c)\big), \cdots \}$ – ербранівський універсум,

$\mathcal{A} = \Big\{ P(c), P\big(f(c)\big), P\Big(f\big(f(c)\big)\Big), P(f\Big(f\big(f(c)\big)\Big), \ldots \Big\}$ – ербранівський базис.

Співставимо константі $c \in H_\infty$ елемент $a \in D$ $(c \longmapsto a \in D)$. Тоді $f(c) \mapsto f(a) = b \in D$, $f\big(f(c)\big) \mapsto f(b) = a \in D$ і т.д. Оскільки $|P(a)| = 0, |P(b)| = 1$, то матимемо наступну H-інтерпретацію $I_H = \{ \neg P(c), P\big(f(c)\big), \neg P\Big(f\big(f(c)\big)\Big), P(f\Big(f\big(f(c)\big)\Big), \ldots \}$, яка відповідає I та множина S виконується на I_H.

Теорема 5.1. (теорема про H-інтерпретації).

Множина диз'юнктів S суперечлива тоді і тільки тоді, коли вона спростовується на кожній своїй H-інтерпретації, тобто на кожній H-інтерпретації знайдеться хоча б один диз'юнкт, який буде хибним хоча б на одному наборі своїх змінних з H_∞.

Доведення.

Необхідність очевидна, оскільки S спростовується на кожній інтерпретації, то S спростовується на кожній H-інтерпретації.

Достатність.

Нехай S виконується на деякій інтерпретації I на предметній області D. Підберемо I_H – H-інтерпретацію, яка відповідає I. Згідно леми 5.4, S виконується на I_H, що і слід було довести.

Кінець доведення.

5.4. Метод семантичних дерев

5.4.1. Поняття семантичного дерева

Будемо працювати з кореневими деревами, можливо з нескінченною, але не більш, ніж зліченою кількістю вершин. При цьому вважатимемо, що степінь будь-якої вершини скінченна. Обмежимося випадком бінарних дерев.

Означення 5.9. Бінарне дерево – це кореневе дерево, у якого всі вершини, крім листів, породжують рівно дві вершини наступного рівня. Приклади бінарних дерев зображені на рисунку 5.1.

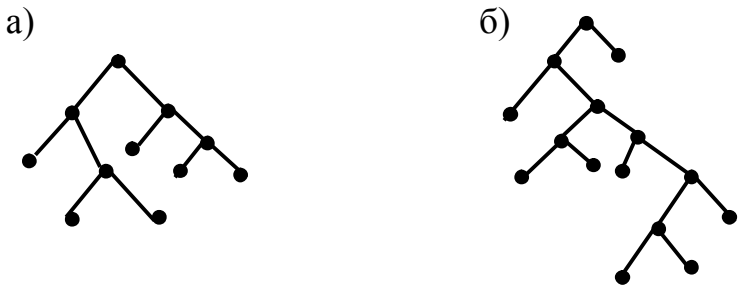


Рис. 5.1. Бінарні дерева

Означення 5.10. Нехай S – множина диз'юнктів.

Семантичним деревом, яке відповідає S, називатимемо бінарне дерево T таке, що:

1) Кожному ребру дерева приписується атом із ербранівського базису, або його заперечення. (P, або $\neg P$, де $P \in \mathcal{A}$);

2) Ребрам, які виходять з однієї вершини (вузла) приписується так звана контрарна пара (P та $\neg P$, рис. 5.2);

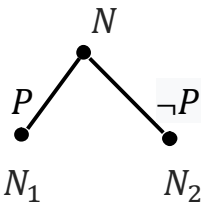


Рис. 5.2.

3) Кожна гілка дерева не містить жодної контрарної пари, а також жоден атом, або його заперечення, не зустрічається на ній більше одного разу.

Означення 5.11. Семантичне дерево T називається повним, якщо кожна його гілка (можливо, нескінченна) містить усі елементи ербранівського базису (сам елемент, або його заперечення).

Приклад 5.14. Нехай

$$S_1 = \begin{Bmatrix} 1)\ P \vee Q, \\ 2)\ \neg Q \vee R, \\ 3)\ \neg P \vee \neg R \end{Bmatrix}.$$

Ербранівський базис

$\mathcal{A} = \{P, Q, R\}$;

Повне семантичне дерево T_1 (рис. 5.3).

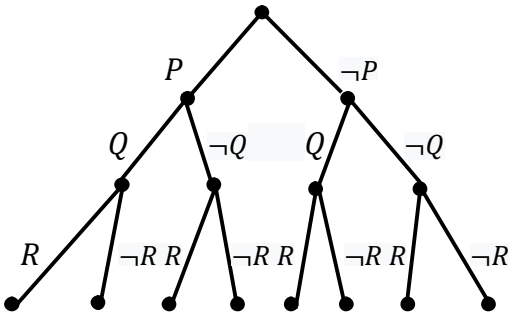


Рис.5.3. Семантичне дерево T_1

Приклад 5.15. Нехай

$S_2 = \left\{ \begin{matrix} 1)P(x) \vee P(a), \\ 2)\neg P(y) \vee \neg P(b) \end{matrix} \right\}$.

Ербранівський універсум

$H_\infty = \{a, b\}$.

Ербранівський базис

$\mathcal{A} = \{P(a), P(b)\}$.

Повне семантичне дерево T_2 (рис. 5.4).

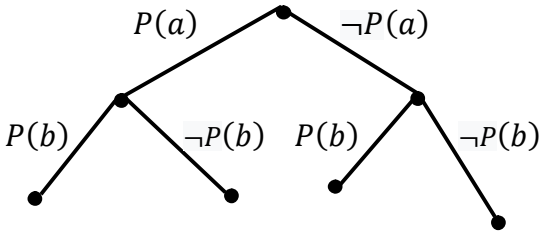


Рис.5.4. Семантичне дерево T_2

Приклад 5.16. Нехай $S_3 = \{1)\ P(x),\ \ 2)\ \neg P(f(y))\}$. Ербранівський універсум $H_\infty = \{a,\ \ f(a),\ \ f(f(a)), \ldots\}$. Ербранівський базис $\mathcal{A} = \{P(a),\ P(f(a)), P(f(f(a))), \ldots\}$ є нескінченним, а тому і повне семантичне дерево T_3 також є нескінченним. (рис. 5.5). Елементи з $\mathcal{A}$ можуть виписуватися в довільному порядку (дерево T_4 рис. 5.6).

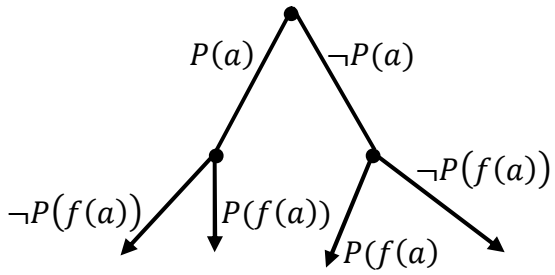


Рис. 5.5. Семантичне дерево T_3

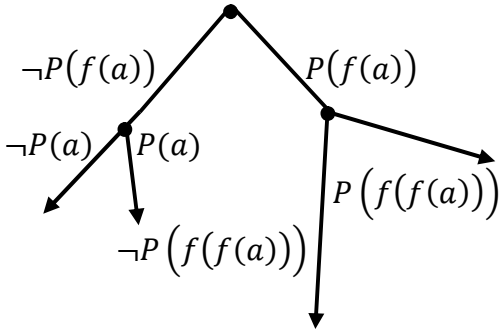


Рис. 5.6. Семантичне дерево T_4

Як бачимо, повному семантичному дереві кожна гілка відповідає деякій H-інтерпретації. Самі H-інтерпретації (часткові H-інтерпретації) визначаються літерами, які приписуються даній гілці (частковій гілці).

Дане твердження справедливе і у зворотній бік. По кожній H-інтерпретації (частковій H-інтерпретації) можна побудувати гілку (часткову гілку) в повному семантичному дереві. Тобто маємо взаємно однозначну відповідність між множиною гілок в повному семантичному дереві та множиною всіх H-інтерпретацій. У випадку нескінченного, але зліченого дерева, потужність множини всіх H-інтерпретацій – континуум.

Означення 5.12. Вузол N семантичного дерева T називається спростовуючим, якщо на відповідній цьому вузлу частковій H-інтерпритації спростовується хоча б один диз'юнкт з S (перетворюється в нуль при деякому наборі значень своїх змінних з H_∞).

Зрозуміло, що всі наступні вузли за N також будуть спростовуючими, а тому для уникнення невизначеності перший такий вузол на гілці вважатимемо спростовуючим, і далі гілку не продовжуємо. Сам спростовуючий вузол позначатимемо символом ✕.

Означення 5.13. Семантичне дерево T називається замкнутим, якщо кожна його гілка закінчується спростовуючим вузлом.

Приклад 5.17. З урахуванням даного факту семантичним деревам $T_1 - T_3$ з прикладів 5.14-5.16 будуть відповідати дерева $T_1' - T_3'$ (рис. 5.7 – 5.9 відповідно).

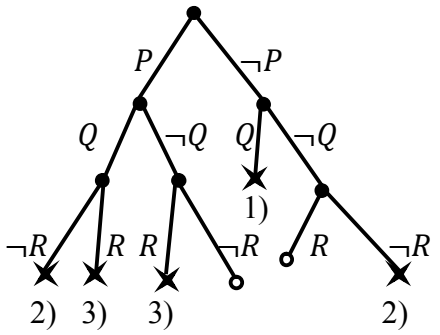


Рис. 5.7. Семантичне дерево T_1'

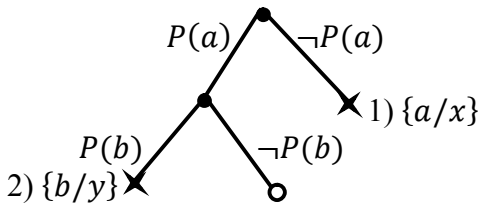


Рис. 5.8. Семантичне дерево T_2'

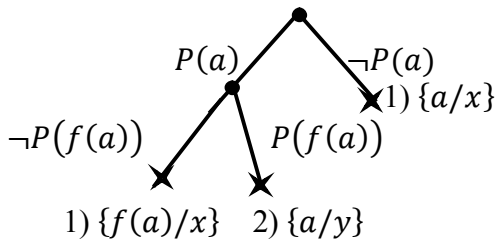


Рис. 5.9. Семантичне дерево T_3'

Тут біля кожного спростовуючого вузла вказується номер диз'юнкта з відповідної множини S та значення змінних, при яких даний диз'юнкт спростовується. Запис $\{t/x\}$ означає, що на місце змінної x підставлено значення терму t з H_∞.

З побудови семантичного дерева випливає, що умова суперечливості множини диз'юнктів еквівалентна умові замкнутості відповідного семантичного дерева. З рис.5.7 – 5.9 видно, що дерева T_1' та T_2' незамкнуті, а дерево T_3' замкнуте. А тому множини диз'юнктів S_1 та S_2 несуперечливі, а S_3 суперечлива.

Результат вище викладених міркувань підсумовується у вигляді теореми Ербрана[5].

5.4.2. Теорема Ербрана

Теорема 5.2. (теорема Ербрана, формулювання I)**.** Множина диз'юнктів S суперечлива тоді і тільки тоді, коли кожному повному семантичному дереву S відповідає скінченне замкнуте семантичне дерево.

[5] Жак Ербран (1908-1931) — французький математик, логік. Відомі дві теореми Ербрана, які відносяться до зовсім різних теорем. Перша — результат дисертаційної роботи з автоматичного доведення теорем, друга — теорема Ербрана-Рібета, використовується в гомологічній алгебрі.

Доведення. Залишається довести скінченність замкнутого семантичного дерева (Лема Кьоніга).

Припустимо, що дане дерево є замкнутим (рис. 5.10), але нескінченним. Позначимо через N_0 – корінь, тоді хоча б одне з піддерев з вершиною N_1 (ліве), або N_2 (праве) є нескінченним. Без втрати загальності виберемо ліве піддерево, яке є нескінченним. Тоді у нього є також два піддерева, хоча б одне з них нескінченне. Знову візьмемо ліве піддерево з коренем $N_{1.1}$. Цей процес можна ввести нескінченно довго, в результаті чого отримаємо нескінченну гілку, яка складається з вузлів N_0, N_1, $N_{1.1}$, $N_{1.1.1}$, ... , серед яких нема жодного спростовуючого вузла. Тому гілка, яка складається з цих вузлів відповідає H-інтерпретації, на якій не спростовується жоден диз'юнкт з S, а тому S є несуперечливою, що суперечить умові теореми.

Кінець доведення.

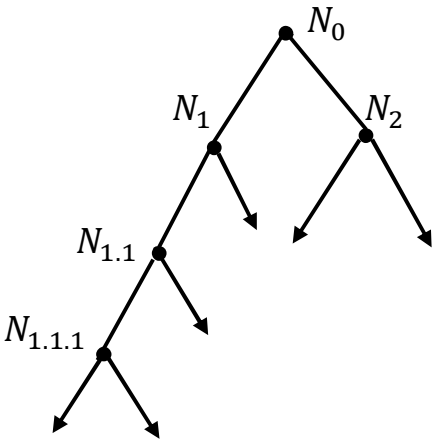


Рис. 5.10. Семантичне дерево для доведення теореми 5.2.

Означення 5.14. Прикладом диз'юнкта $D \in S$ називають диз'юнкт D' отриманий заміною змінних на терми: зміні, константи, або функціональні символи з S.

Тут різні змінні можна замінити на однакові терми.

Означення 5.15. Основним прикладом диз'юнкта $D \in S$ називають приклад D', який не містить змінних.

Фактично всі змінні в D замінено на терми з ербанівського універсуму H_∞.

Приклад 5.18. Для диз'юнкта $D = P\big(x, f(y)\big)$ прикладами є диз'юнкти $D_1 = P\big(x, f(x)\big)$, $D_2 = P\big(f(u), f(u)\big)$, $D_3 = P\Big(z, f\big(f(z)\big)\Big)$, $D_4 = P\big(a, f(a)\big)$, $D_5 = P\Big(f\big(f(a)\big), f(a)\Big)$, з яких D_4 та D_5 є основними.

Теорема 5.3 (Теорема Ербрана, формулювання II) Множина диз'юнктів S суперечлива тоді і тільки тоді, коли існує скінченна суперечлива підмножина основних прикладів диз'юнктів з S. ($\exists S_0 \subset S$, S_0 – скінченна, суперечлива та містить лише основні приклади диз'юнктів з S).

Доведення. Твердження даної теореми безпосередньо випливає з теореми 5.2. Для суперечливої множини S будуємо замкнуте скінченне семантичне дерево, що містить скінченну кількість спростовуючих вузлів. А кожному такому вузлу відповідає основний приклад диз'юнкта, що спростовується на відповідній частковій H-інтерпретації. Множина всіх таких основних прикладів диз'юнктів утворює шукану множину S_0.

Кінець доведення.

Приклад 5.19. Довести суперечливість множини диз'юнктів

$$S = \big\{1)\ P(x, a) \vee Q\big(f(x)\big), 2)\ \neg P(f(x), y) \vee Q(z), 3)\ \neg Q(y)\big\}.$$

Будуємо ербранівський універсум

$H_\infty = \{a, f(a), f(f(a)), \dots\}$.

Можна полічити, що в ербранівському базисі $\mathcal{A}$ доцільно в першу чергу використати елементи $P(f(a), a)$ та $Q\left(f(f(a))\right)$. Тобто $\mathcal{A} = \{P(f(a), a), Q\left(f(f(a))\right), \dots\}$.

Почнемо побудову дерева з $Q\left(f(f(a))\right)$ (див. рис 5.11).

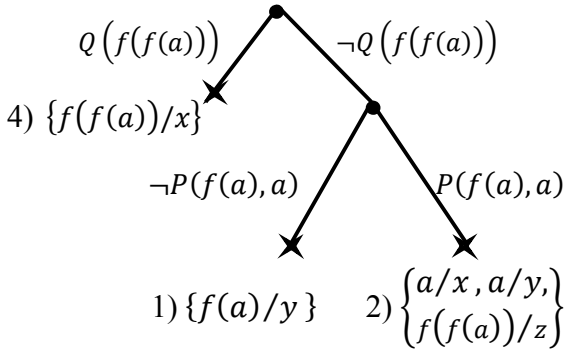


Рис.5.11. Семантичне дерево для прикладу 5.19

Як бачимо, дане повне семантичне дерево є скінченним замкнутим, а тому задана множина диз'юнктів S є суперечливою.

5.4.3. Використання методу семантичних дерев для доведення логічного наслідку.

Задача. Довести логічний наслідок $A_1, A_2, \dots, A_n \vDash B$, де A_i, $i = \overline{1,n}$ та B – замкнуті формули в алгебрі предикатів..$n \in N \cup \{0\}$. За теоремою дедукції (Теорема 4.1) умова існування логічного наслідку еквівалентна тому, що формула $(A_1 \wedge A_2 \wedge .. \wedge A_n) \to B$ є ЛЗЗ, а тому формула $A_1 \wedge A_2 \wedge .. \wedge A_n \wedge \neg B$ є суперечністю. Замість зведення до ССФ всієї формули, зведемо до ССФ кожну формулу A_i, $i = \overline{1,n}$ та $\neg B$ окремо (див. формулу 4 в пункті 4.3.1.) Отримаємо множини диз'юнктів

$S_1, S_2, .. S_n$ та S_{n+1} відповідно. Тоді задача зведеться до доведення суперечливості множини $S = \bigcup_{i=1}^{n+1} S_i$.

Зауваження 5.5. Цей метод доведення логічного наслідку зручно використовувати також у методі резолюцій (див. параграф 5.5.).

Приклад 5.20. Методом семантичних дерев довести логічний наслідок для задачі з прикладу 4.19. Деякі студенти люблять своїх викладачів. Жоден студент не любить невігласів. Отже, жоден викладач не є невігласом.

Гіпотези:

F_1: $\exists x \left(S(x) \wedge \forall y\big(T(y) \to L(x,y)\big)\right)$;

F_2: $\forall x \left(S(x) \to \forall y\big(Q(y) \to \neg L(x,y)\big)\right)$.

Наслідок:

G: $\forall y \left(T(y) \to \neg Q(y)\right)$.

Зведемо формулу F_1 до ССФ.

$$F_1 \Leftrightarrow \exists x \left(S(x) \wedge \forall y\big(T(y) \to L(x,y)\big)\right) \Leftrightarrow$$

$$\Leftrightarrow \exists x \forall y \left(S(x) \wedge \left(\neg T(y) \vee L(x,y)\right)\right) \rightsquigarrow$$

$$\rightsquigarrow \forall y \left(S(a) \wedge \left(\neg T(y) \vee L(a,y)\right)\right) - \text{ССФ}.$$

Отримали множину S_1, яка містить два диз'юнкти.

$$S_1 = \{S(a), \neg T(y) \vee L(a,y)\}$$

Перейдемо до формули F_2.

$$F_2 \Leftrightarrow \forall x \left(\neg S(x) \vee \forall y\big(\neg Q(y) \vee \neg L(x,y)\big)\right) \Leftrightarrow$$

$$\Leftrightarrow \forall x \forall y\big(\neg S(x) \vee \neg Q(y) \vee \neg L(x,y)\big) - \text{ССФ}.$$

В S_2 маємо один диз'юнкт. $S_2 = \{\neg S(x) \vee \neg Q(y) \vee \neg L(x,y)\}$.

Розглянемо заперечення до G.

$$\neg G \Leftrightarrow \neg \forall y\big(T(y) \to \neg Q(y)\big) \Leftrightarrow \exists y\big(T(y) \wedge Q(y)\big) \rightsquigarrow T(b) \wedge Q(b) - \text{ССФ}.$$

Далі доведемо суперечливість множини

$$S = S_1 \cup S_2 \cup S_3 = \begin{Bmatrix} 1)\ S(a), 2)\ \neg T(y) \vee L(a,y), \\ 3)\ \neg S(x) \vee \neg Q(y) \vee \neg L(x,y), 4)\ T(b),\ \ 5) Q(b) \end{Bmatrix}.$$

Ербранівський універсум $H_\infty = \{a, b\}$.

Ербранівський базис $\mathcal{A} = \{S(a), T(b), Q(b), L(a,b), \dots\}$.

Будуємо семантичне дерево (рис.5.12).

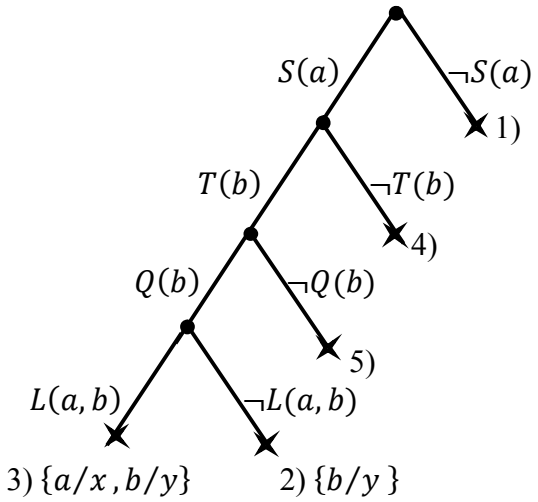


Рис.5.12. Семантичне дерево для прикладу 5.20

Логічний наслідок доведено.

5.5. Метод резолюцій

Розглянемо інший метод автоматичного доведення теорем – метод резолюцій. Наведений в попередньому параграфі метод семантичних дерев має істотний недолік, який полягає в тому, що потужність ербранівського універсуму, а разом з ним потужність ербранівського базису та множини основних прикладів диз'юнктів із S в багатьох випадках зростає

експоненціально (див. приклад 5.8), що суттєво ускладнює реалізацію його на ЕОМ.

Запропонований Дж. Робінсоном[6] метод резолюцій застосовувати можна для довільної множини диз'юнктів з метою перевірки її суперечливості без знаходження ербранівського універсуму та базису.

Основна ідея цього методу полягає в тому, щоб перевірити наявність порожнього диз'юнкту (□) в S. Як було сказано в параграфі 5.2., якщо $\square \in S$, то S суперечлива. В іншому випадку намагаємося отримати □ з S шляхом дописування нових формул без втрати суперечливості (несуперечливості) вихідної множини диз'юнктів.

5.5.1. Лема про резолюцію

Теорема 5.4. (лема про резолюцію). Нехай множина диз'юнктів S містить два диз'юнкти $D_1 = A_1 \vee A_2 \vee \ldots \vee A_n \vee C$ та $D_2 = B_1 \vee B_2 \vee \ldots \vee B_m \vee \neg C$, де A_i та B_j – літери, $i = \overline{1,n}$, $j = \overline{1,m}$, $m, n \geq 0$, а C – атом. Тоді без втрати суперечливості (несуперечливості) до множини S можна дописати диз'юнкт

$D = A_1 \vee A_2 \vee \ldots \vee A_n \vee B_1 \vee B_2 \vee \ldots \vee B_m$.

Доведення. Якщо множина S суперечлива, то при дописуванні до неї будь-яких диз'юнктів суперечливість не порушиться. Якщо ж S

[6] Джон Алан Робінсон (1930 – 2016) – англійський філософ та логік, зробив важливий внесок у становлення логічного програмування. В 1965 році опублікував роботу «Машинно-орієнтована логіка, заснована на принципі резолюцій», яка є основоположною в автоматизації правила резолюції в логіці. Його роботи стали вирішальними в розвитку мови програмування Пролог.

несуперечлива, то твердження теореми випливає з логічного наслідку $D_1, D_2 \vDash D$.

Вправа 5.3. Довести логічний наслідок $D_1, D_2 \vDash D$.

Кінець доведення.

Розглянемо два частинні випадки:

$m = 0, n \neq 0$ $A_1 \vee A_2 \vee \ldots \vee A_n \vee C, \neg C \vDash A_1 \vee A_2 \vee \ldots \vee A_n$,;

$m = n = 0, C, \neg C \vDash \square$.

За допомогою теореми 5.4. можна доводити суперечливість множини диз'юнктів в алгебрі висловлень.

Для алгебри висловлень введено поняття резольвенти та резольвентного виведення.

Означення 5.16. Диз'юнкт D, що фігурує в формулюванні теореми 5.4. називатимемо резольвентою диз'юнктів D_1та D_2 і позначатимемо $D = = R(D_1, D_2)$.

Означення 5.17. Резольвентним виведенням диз'юнкта D з множини диз'юнктів S називається скінченна послідовність диз'юнктів $D_1, D_2, \ldots, D_n$, $n \in \mathbb{N}$, кожен з яких або належить S, або є резольвентою диз'юнктів, які йому передують, і $D_n = D$.

Тоді для доведення суперечливості S достатньо побудувати резольвентне виведення $\square$ з S. Те, що для суперечливої множини диз'юнктів таке виведення завжди існує буде доведено в 5.5.4.

Приклад 5.21. Нехай $S = \{P \vee \neg Q, \neg P, Q\}$. Доведемо її суперечливість:

1 $P \vee \neg Q$

2 $\neg P$

3 Q

---

4 $\neg Q$ $R(1,2)$

5 $\square$ $R(3,4)$

Приклад 5.22. Нехай $S = \{P \vee Q, P \vee \neg Q, \neg P \vee Q, \neg P \vee \neg Q\}$.

Будуємо резольвентне виведення $\square$:

1 $P \vee Q$

2 $P \vee \neg Q$

3 $\neg P \vee Q$

4 $\neg P \vee \neg Q$

---

5 P $R(1,2)$

6 $\neg P$ $R(3,4)$

7 $\square$ $R(5,6)$

Таким чином довели, що множина диз'юнктів в S є суперечливою.

Для того, щоб доводити суперечливість множини диз'юнктів, в алгебрі предикатів нам знадобиться поняття підстановки та уніфікації.

5.5.2. Підстановки та уніфікація

Твердження 5.1. В першу чергу відмітимо те, що до множини диз'юнктів без втрати суперечливості (несуперечливості) можна дописувати будь-який приклад будь-якого диз'юнкта. Обґрунтування цього факту випливає з теореми 5.3, оскільки множина основних прикладів диз'юнктів від цього не зміниться.

Приклад 5.23. Розглянемо два диз'юнкти $D_1 = P(x) \vee Q(x)$ та $D_2 = \neg P\big(f(x)\big) \vee R(x)$. Контрарних пар літер одна з D_1, а інша з D_2 не існує. однак, якщо ми замінимо x на $f(a)$ в D_1 і x на a в D_2, то отримаємо диз'юнкти $D'_1 = P\big(f(a)\big) \vee Q\big(f(a)\big)$ та $D'_2 = \neg P\big(f(a)\big) \vee R(a)$, які можна дописати до S без втрати суперечливаості (несуперечливості). Ці диз'юнкти мають контрарну пару $P\big(f(a)\big)$ та $\neg P\big(f(a)\big)$ і для них існує резольвентна $D = Q\big(f(a)\big) \vee R(a)$. І таких замін можна зробити багато. Наприклад, в D_1 x замінити на $f\big(f(a)\big)$, а в D_2 x – на $f(a)$, або в D_1 x замінити на $f(x)$, або в D_1 x замінити на $f(y)$, а в D_2 x – на y.

Означення 5.18. Виразом будемо називати або формулу в алгебрі предикатів, яка не містить кванторів, або терм.

Через $E(x_1, x_2, \ldots, x_n)$ позначимо вираз, який містить змінні $x_1, x_2, \ldots, x_n$, $n \geq 0$.

Означення 5.19. Підстановкою σ на множині виразів називається скінченна множина вигляду $\sigma = \{t_1/x_1, t_2/x_2, \ldots, t_n/x_n\}$, де x_i, $i = \overline{1,n}$ – змінні, які попарно відрізняються, t_i – терми, відмінні від x_i для всіх $i = \overline{1,n}$. Дією підстановки σ на вираз E називатимемо операцію одночасної заміни всіх входжень змінної x_1 на t_1, x_2 на t_2, ... , x_n на t_n і позначатимемо $E\sigma$. Підстановка, яка не містить елементів називається порожньою і позначається ε, або { }. $E\varepsilon = E$.

Приклад 5.24. $P\big(x, f(y)\big)\{f(y)/x, f(x)/y\} = P\big\{f(y), f\big(f(x)\big)\big\}$.

$$g(x, f(x), y)\{f(y)/x, f\,(x)/y\} = g\big\{f(y), f\big(f(y)\big), f(x)\big\}.$$

Означення 5.20. На множині підстановок вводиться операція композиції $E(\sigma_1 \circ \sigma_2) = (E\sigma_1)\sigma_2$.

Приклад 5.25. Нехай $\sigma_1 = \{a/x, f(z)/y\}$, $\sigma_2 = \{c/z\}$. Тоді $\sigma_1 \circ \sigma_2 = \{a/x, f(c)/y, c/z\}$, $\sigma_2 \circ \sigma_1 = \{c/z, a/x, f(c)/y\} \neq \sigma_1 \circ \sigma_2$.

Як бачимо, операція $\circ$ некомутативна.

Вправа 5.4. Довести, що множина підстановок відносно операції $\circ$ є моноїдом з нейтральним елементом ε.

Означення 5.21. Підстановка σ називається уніфікатором для множини виразів $\{E_1, E_2, \ldots, E_k\}$ тоді і тільки тоді, коли $E_1\sigma = E_2\sigma = \ldots = E_k\sigma$. Якщо для $\{E_1, E_2, \ldots, E_k\}$ існує хоча б один уніфікатор, то ця множина називається уніфікованою.

Приклад 5.26. Для виразів $P(x, a)$, $P(b, y)$ уніфікатором буде $\sigma = = \{b/x, a/y\}$, для виразів $g(x)$ та $g\big(f(y)\big)$ є безліч уніфікаторів, наприклад, $\sigma_1 = \{f(a)/x, a/y\}$, $\sigma_2 = \{f\,\big(g(x)\big)/x, g(x)/y\}$, $\sigma_3 = \{f\,(y)/x\}$ тощо. Уніфікатор для виразів існує далеко не завжди, наприклад, $P(x, a)$ та $P(b, x)$ не уніфікується, оскільки неможливо поміняти змінну x на константи a та b одночасно. Також не уніфікуються вирази $g(x)$ та $g\big(f(x)\big)$ тому, що при заміні x на терм t отримаємо $g(t)$ та $g\big(f(t)\big)$.

Можливий випадок, коли для виразів E_1 та E_2 існують підстановки σ_1 та σ_2 такі, що $E_1\sigma_1 = E_2$ та $E_2\sigma_2 = E_1$. Це можливо тоді і тільки тоді, коли вирази відрізняються лише назвами змінних. Такі вирази називаються еквівалентними ($E_1 \sim E_2$). Як правило, еквівелентні вирази не розрізняють.

Приклад 5.27. Нехай $P(x, y) \sim P(y, x) \sim P(u, v)$, але $P(x, y) \nsim P(x, x)$.

Означення 5.22. Уніфікатор σ для множини виразів $\{E_1, E_2, \ldots, E_k\}$ називається найбільшим спільним уніфікатором (НСУ) тоді і тільки тоді, коли для будь-якого уніфікатора σ_1 знайдеться підстановка σ_0 така, що $\sigma_1 = \sigma \circ \sigma_0$.

Приклад 5.28. Для виразів $P(x, a)$ та $P(b, y)$ уніфікатор $\sigma = \{b/x, a/y\}$ є єдиним, а тому він буде НСУ. А для виразів $g(x)$ та $g\big(f(x)\big)$ НСУ буде $\sigma = \{f(y)/x\}$ тому, що будь-який уніфікатор

$\sigma_1 = \{f(t)/x, t/y\}$, де t — довільний терм з S можна подати у вигляді $\sigma_1 = \sigma \circ \{t/y\}$.

Далі наведемо алгоритм уніфікації для пошуку НСУ для скінченної множини уніфікованих виразів. Цей алгоритм також визначає факт того, що множина виразів не уніфікується.

Алгоритм уніфікації

Маємо множину виразів $W = \{E_1, E_2, \ldots, E_m\}$.

Крок 1. Покладемо $k = 0$, $W_k = W$ та $\sigma_k = \varepsilon$.

Крок 2. Якщо W містить один вираз, то зупиняємося: σ_k – НСУ для W. В іншому випадку знаходимо першу зліва позицію, на якій є невідповідність хоча б в одному символі хоча б для двох виразів з W_k. Множину таких символів позначимо через D_k.

Крок 3. Якщо існує змінна x_k та терм t_k, який не містить x_k в D_k, то переходимо на крок 4, в іншому випадку зупинка: W не уніфіковується.

Крок 4. Нехай $\sigma_{k+1} = \sigma_k \circ \{t_k/x_k\}$ і $W_{k+1} = W_k\{t_k/x_k\}$ ($W_{k+1} = W\sigma_{k+1}$).

Крок 5. Збільшуємо k на одиницю та переходимо на крок 2.

Даний алгоритм закінчує роботу за скінченну кількість кроків тому, що вихідна множина W містить скінченну кількість змінних, а кожен крок алгоритму зменшує кількість змінних на одиницю (пропадає x_k). Обгрунтування того, що алгоритм дозволяє будувати саме НСУ для множини виразів див. у [3, с. 84].

Приклад 5.29. Знайти НСУ для

$$W = \{P(f(x,y), g(y)), P(f(a, g(x)), z)\}.$$

Нехай $\sigma_0 = \varepsilon$, $W_0 = W$ і потужність W_0 більше, ніж один ($|W_0| > 1$), тому σ_0 – не НСУ для W.

Множина невідповідності $D_0 = \{x, a\}$ містить змінну $x_0 = x$ та терм $t_0 = a$, який не містить x_0.

Нехай $\sigma_1 = \sigma_0 \circ \{t_0/x_0\} = \varepsilon \circ \{a/x\} = \{a/x\}$,

$$W_1 = W_0\{t_0/x_0\} = \left\{P\big(f(x,y), g(y)\big)\{a/x\}, P\left(f\big(a, g(x)\big)\right), z\{a/x\}\right\} =$$
$$= \{P\big(f(a,y), g(y)\big), P\big(f\big(a, g(a)\big), z\big)\}.$$

Оскільки $|W_1| > 1$, будуємо $D_1 = \{y, g(a)\}$, яка містить терм $g(a) = t_1$, який не містить змінну $x_1 = y$.

$$\sigma_2 = \sigma_1 \circ \{t_1/x_1\} = \{a/x\} \circ \{g\,(a)/y\} = \{a/x\,, g\,(a)/y\},$$
$$W_2 = W_1\{t_1/x_1\} = \{P\big(f(a,y), g(y)\big), P\big(f\big(a, g(a)\big), z\big)\}\{g\,(a)/y\} =$$
$$= \left\{P\left(f\big(a, g(a)\big), g\big(g(a)\big)\right), P\big(f\big(a, g(a)\big), z\big)\right\}.$$

Знову $|W_2| > 1$, будуємо $D_2 = \{g\big(g(a)\big), z\}$, яка містить терм $t_2 = g\big(g(a)\big)$ в записі якого не міститься змінна $x_2 = z$.

$$\sigma_3 = \sigma_2 \circ \{t_2/x_2\} = \{a/x\,, g\,(a)/y\,, g\left(g(a)\right)/z\},$$
$$W_3 = W_2\{t_2/x_2\} = \left\{P\left(f\big(a, g(a)\big), g\big(g(a)\big)\right), P\big(f(a), g(a)\big), z\right\}\{g\left(g(a)\right)/z\} =$$
$$= \left\{P\left(f\big(a, g(a)\big), g\big(g(a)\big)\right)\right\}.$$

Оскільки $|W_3| = 1$, то $\sigma_3 = \{a/x\,, g\,(a)/y\,, g\left(g(a)\right)/z\}$ – НСУ для W.

Кінець.

Приклад 5.30 Покажемо, що множина виразів $W = \{Q\big(x, g(x)\big), Q\big(a, f(y)\big)\}$ не уніфікується.

Нехай $\sigma_0 = \varepsilon$, $W_0 = W$.

$|W_0| > 1$, $D_0 = \{x, a\}$ містить змінну $x_0 = x$ та терм $t_0 = a$, який не містить x_0.

$$\sigma_1 = \sigma_0 \circ \{t_0/x_0\} = \varepsilon \circ \{a/x\} = \{a/x\},$$
$$W_1 = W_0\{t_0/x_0\} = \{Q\big(x, g(x)\big), Q\big(a, f(y)\big)\} \circ \{a/x\} =$$
$$\{Q\big(a, g(a)\big), Q\big(a, f(y)\big)\}.$$

$|W_1| > 1$, $D_1 = \{g(a), f(y)\}$. Оскільки D_1 не містить змінної, то приходимо до висновку, що W не уніфікується.

Кінець.

5.5.3. Метод резолюцій: загальний випадок

Означення 5.23. Нехай множина диз'юнктів S містить два диз'юнкта $D_1 = A_1 \vee A_2 \vee ... \vee A_n \vee C_1$ та $D_2 = B_1 \vee B_2 \vee ... \vee B_m \vee \neg C_2$, $n, m \geq 0$, які не містять спільних змінних. Тут A_i, $i = \overline{1,n}$ та B_j, $j = \overline{1,m}$ – літери, а C_1 та C_2 – атоми, які мають НСУ σ ($C_1\sigma = C_2\,\sigma$). Тоді диз'юнкт вигляду $A_1\sigma \vee A_2\sigma \vee ... \vee A_n\sigma \vee B_1\sigma \vee B_2\sigma \vee ... \vee B_m\sigma$ називається бінарною резольвентою диз'юнктів D_1 та D_2. У цьому випадку говорять, що літери C_1 та C_2 відрізаються.

Зрозуміло, що бінарну резольвенту можна дописувати до S із збереженням суперечливості, або несуперечливості. У випадку, коли диз'юнкти мають спільні змінні, користуючись зауваженням 5.3 перейменовуємо змінні.

Приклад 5.31. Нехай множина диз'юнктів S містить такі диз'юнкти: $D_1 = P(x, a) \vee \neg Q\big(f(y)\big)$, $D_2 = \neg P(b, y) \vee \neg R(x)$, $D_3 = Q(x) \vee R\big(f(x)\big)$. Згідно з зауваженням 5.3 перейменуємо змінні: в D_1 – x на x_1 та y на y_1, в D_2 – x на x_2 та y на y_2 і в D_3 – x на x_3. $D_1 = P(x_1, a) \vee \neg Q\big(f(y_1)\big)$, $D_2 = \neg P(b, y_2) \vee \neg R(x_2)$, $D_3 = Q(x_3) \vee R\big(f(x_3)\big)$. Диз'юнкти D_1 та D_2 мають контрарну пару літер P та $\neg P$. Вирази $P(x_1, a)$ та $P(b, y_2)$ уніфікуються, НСУ $\sigma = \{b/x_1\,, a/y_2\}$, тому їх бінарна резольвента буде

$\neg Q\big(f(y_1)\big) \vee \neg R(x_2)$. Змінні y_1 та x_2 ми тут не чіпали. У диз'юнктів D_1 та D_3 за допомогою НСУ $\sigma = \{f(y_1)/x_3\}$ ототожнюються літери $Q(x_3)$ та $Q(f(y_1))$, а тому їх бінарна резольвента буде виглядати так: $P(x_1, a) \vee R\Big(f\big(f(y_1)\big)\Big)$. І, нарешті, у D_2 та D_3 можна відрізати літеру R. НСУ для $R(x_2)$ та $R\big(f(x_3)\big)$ є $\sigma = \{f(x_3)/x_2\}$ і бінарна резольвента D_2 та D_3 дорівнює $\neg P(b, y_2) \vee Q(x_3)$.

Але навіть бінарної резольвенти може не вистачати для доведення суперечливості множини диз'юнктів S.

Приклад 5.32. Нехай $S = \{P(x_1) \vee P(x_2), \neg P(y_1) \vee \neg P(y_2)\}$. Дана множина диз'юнктів буде суперечливою, що легко доводиться, наприклад, за допомогою методу семантичних дерев.

1. $P(x_1) \vee P(x_2)$;
2. $\neg P(y_1) \vee \neg P(y_2)$.

Бінарна резольвента для диз'юнктів 1 та 2 може записуватися у вигляді:

3. $P(x_1) \vee \neg P(y_2)$ $\quad R(1,2)\ \{x_1/y_1\}$.

Всі інші спроби взяття бінарної резольвенти з 1, 2, 3 дадуть еквівалентні вище написаним диз'юнкти. Не забуваймо, що згідно з означенням 5.23 диз'юнкти не повинні містити спільних змінних.

Щоб уникнути такої проблеми, вводиться поняття склейки.

Означення 5.24. Якщо дві, або більше літер (з однаковим знаком) диз'юнкта D мають НСУ σ, то диз'юнкт $D\sigma$ називається склейкою D.

Приклад 5.33. Для диз'юнкта $D = P(x) \vee P\big(f(y)\big) \vee \neg Q\big(g(x, y)\big)$ перша та друга літери мають НСУ $\sigma = \{f(y)/x\}$, тому $D\sigma = P\big(f(y)\big) \vee \neg Q\big(g(f(y), y)\big)$ є склейкою D.

Приклад 5.34. Із диз'юнкта $D = \neg P(x) \vee \neg P\big(f(y)\big) \vee \neg P\big(f(a)\big)$ можна отримати три різні склейки:

1) уніфікуємо першу та другу літери D ($\sigma_1 = \{f(y)/x\}$), або першу та третю літери ($\sigma_2 = \{f(a)/x\}$) отримаємо склейку $D\sigma_1 = D\sigma_2 = \neg P\big(f(y)\big) \vee \neg P\big(f(a)\big)$;
2) уніфікуємо другу та третю літери ($\sigma_3 = \{a/y\}$) і отримаємо склейку $D\sigma_3 = \neg P(x) \vee \neg P\big(f(a)\big)$;
3) уніфікуємо всі три літери в D ($\sigma_4 = \{f(a)/x,\ a/y\}$) і отримуємо склейку $D\sigma_4 = \neg P\big(f(a)\big)$.

Означення 5.25. (означення резольвенти для алгебри предикатів). Нехай D_1 та D_2 – деякі диз'юнкти з множини диз'юнктів S, тоді їх резольвентою називатимемо один з наступних чотирьох диз'юнктів:

1. Бінарна резольвента D_1 та D_2;
2. Бінарна резольвента D_1 та склейки D_2;
3. Бінарна резольвента склейки D_1 та D_2;
4. Бінарна резольвента склейки D_1 та склейки D_2.

Означення резольвентного виведення (означення 5.17) буде справедливе і в цьому випадку.

Приклад 5.35. Для диз'юнктів $D_1 = P(x_1) \vee P(x_2)$ та $D_2 = \neg P(y_1) \vee \neg P(y_2)$ (див. приклад 5.32) склейками будуть диз'юнкти $D_1' = P(x_1)$ ($\sigma_1 = \{x_1/x_2\}$) та $D_2' = \neg P(y_1)$ ($\sigma_2 = \{y_1/y_2\}$) відповідно і їх бінарною резольвентою буде $\square$ ($\sigma_3 = \{x_1/y_1\}$). Тобто множина $S = \{D_1, D_2\}$ є суперечливою.

Записувати це резольвентне виведення будемо в такому вигляді:

1 $P(x_1) \vee P(x_2)$

2 $\neg P(y_1) \vee \neg P(y_2)$

3 □ $R(1\,\{x_1/x_2\},\ 2\,\{y_1/y_2\})\,\{x_1/y_1\}$.

Наведемо кілька прикладів використання методу резолюції.

Приклад 5.36. Довести суперечливість множини диз'юнктів

$$S = \{P(x) \vee P\big(f(a)\big) \vee \neg Q\big(y, f(b)\big),\ \neg P\big(f(x)\big) \vee \neg Q(x, y),$$
$$Q\big(x, f(y)\big) \vee Q(a, z)\}.$$

Випишемо диз'юнкти, попередньо перейменувавши в них змінні, та побудуємо резольвентне виведення □.

1 $P(x_1) \vee P\big(f(a)\big) \vee \neg Q\big(y_1, f(b)\big)$

2 $\neg P\big(f(x_2)\big) \vee \neg Q(x_2, y_2)$

3 $Q\big(x_3, f(y_3)\big) \vee Q(a, z)$

4 $\neg P\big(f(x_2)\big)$ $R(2, 3\,\{a/x_3\,, f(y_3)/z\})\,\{a/y_1,\ b/y_2\}$

5 $\neg Q\big(y_1, f(b)\big)$ $R(1\,\{f(a)/x_1\}, 4)\,\{\,\}$

6 □ $R(3\,\{a/x_3\,, f(y_3)/z\}, 5)\{a/y_1,\ b/y_3\}$.

Приклад 5.37. (Продовження прикладів 4.13 та 5.20). Довести логічний наслідок методом резолюцій: Деякі студенти люблять своїх викладачів. Жоден студент не любить невігласів. Отже, жоден викладач не є невігласом. Раніше ми отримали множину диз'юнктів $S = \{S(a),\ \neg T(y) \vee L(a, y),\ \neg S(x) \vee \neg Q(y) \vee \neg L(x, y),\ T(b),\ Q(b)\}$. Логічний наслідок еквівалентний суперечливості цієї множини диз'юнктів (див. пункт 5.4.3 та зауваження 5.5).

Виписуємо диз'юнкти та будуємо резольвентне виведення □.

1	$S(a)$	
2	$\neg T(y_1) \vee L(a, y_1)$	
3	$\neg S(x) \vee \neg Q(y_2) \vee \neg L(x, y_2)$	
4	$T(b)$	
5	$Q(b)$	
	_____	
6	$L(a, b)$	$R(2, 4)\ \{b/y_1\}$
7	$\neg S(a) \vee \neg Q(b)$	$R(3, 6)\ \{a/x, b/y_2\}$
8	$\neg Q(b)$	$R(1,7)\ \{\}$
9	□	$R(5, 8)\ \{\}$.

Логічний наслідок доведено.

У наступному пункті покажемо, що до суперечливої множини диз'юнктів S резольвентне виведення порожнього диз'юнкта завжди існує.

5.5.4. Повнота методу резолюцій

Є тісний зв'язок між семантичним деревом та резольвентним виведенням. Проілюструємо це на прикладі.

Приклад 5.38. Розглянемо множину диз'юнктів $S = \{1)\ \neg P,\ 2)\ P \vee Q, 3)\ P \vee \neg Q\}$.

Ербранівський базис S $\mathcal{A} = \{P, Q\}$. Множина S відповідає замкнутому семантичному дереву T (див. рис. 5.13 а)).

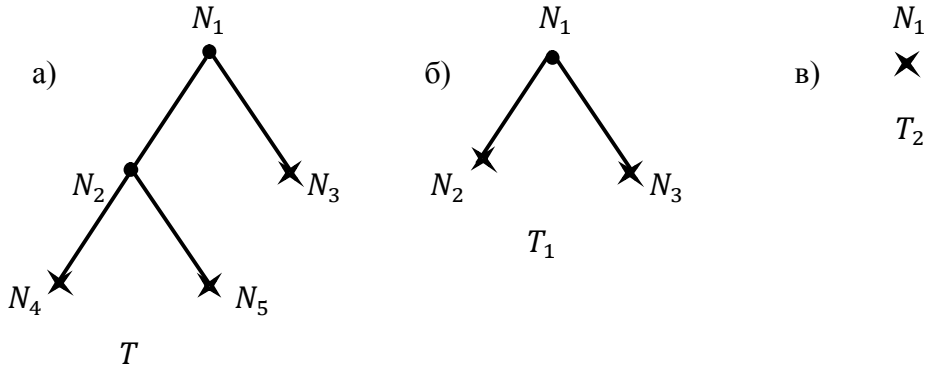


Рис. 5.13 Семенетичні дерева для прикладу 5.38

У дереві T є вузол N_2, який породжує два листи (спростовуючі вузли N_4 та N_5). Вузол N_4 спростовує третій, а N_5 – другий диз'юнкт з S. Беручи резольвенту від цих двох диз'юнктів отримаємо диз'юнкт 4) P, який допишемо до S. Маємо нову множину диз'юнктів $S_1 = S \cup \{P\}$, якій відповідає замкнуте семантичне дерево T_1 (рис. 5.13 б)). В T_1 є вузол N_1, який породжує спростовуючі вузли N_2 та N_3 для диз'юнктів P та $\neg P$ відповідно. Знову візьмемо резольвенту $R(1,4)$, отримаємо порожній диз'юнкт, який допишемо до S_1. $S_2 = S_1 \cup \{\square\} = \{\neg P,\ P \vee Q,\ P \vee \neg Q, P, \square\}$. Семантичне дерево T_2 для S_2 зображене на рис. 5.13 в).

Дана процедура «стягування» семантичного дерева насправді відповідає резольвентному виведенню:

1. $\neg P$
2. $P \vee Q$
3. $P \vee \neg Q$
4. $\overline{\qquad\qquad\qquad}$
   P $\qquad\qquad$ $R(2,3)$
5. $\square$ $\qquad\qquad$ $R(1,4)$

Ідею, викладену в прикладі 5.38, використаємо для доведення повноти методу резолюції, але спочатку доведемо лему, яку називають лемою про підйом.

Лема 5.5. (лема про підйом).

Нехай диз'юнкти $D_1^{'}$ та $D_2^{'}$ – приклади диз'юнктів D_1 та D_2 відповідно, і нехай $D^{'}$ – резольвенти $D_1^{'}$ та $D_2^{'}$. Тоді для D_1 та D_2 існує така резольвента D, що $D^{'}$ є прикладом D.

Доведення

Якщо D_1 та D_2 мають спільні змінні, то перейменуємо їх. Нехай $D_1^{'} = D_1\theta_1$ і $D_2^{'} = D_2\theta_2$ – приклади D_1 та D_2 відповідно. Підберемо підстановки θ_1 та θ_2 так, щоб у них змінні не повторювалися. Позначимо $\theta = \theta_1 \cup \theta_2$. Бачимо, що $D_1^{'} = D_1\theta$, $D_2^{'} = D_2\theta$.

Запишемо диз'юнкти D_1 та D_2 у вигляді: $D_1 = A \vee L_1$, $D_2 = B \vee L_2$, де $A = \bigvee_{i=1}^{n} A_i$, $B = \bigvee_{j=1}^{m} B_j$, A_i B_j – літери, $i = \overline{1,n}$, $j = \overline{1,m}$, $n, m \geq 0$. $L_1 = \bigvee_{l=1}^{r} L_1^l$, $L_2 = \bigvee_{p=1}^{s} (\neg L_2^p)$, де L_1^l та L_2^p – атоми, $l = \overline{1,r}$, $p = \overline{1,s}$, $r, s \geq 1$. L_1 та L_2 вибрані таким чином, що при знаходженні резольвенти $R(D_1^{'}, D_2^{'})$, $L_1\theta$ та $L_2\theta$ відріжуться.

Розпишемо процедуру знаходження $R(D_1^{'}, D_2^{'})$ докладніше. $D_1^{'} = A\theta \vee L_1\theta$, $D_2^{'} = B\theta \vee L_2\theta$. Якщо необхідно, знайдемо склейку диз'юнктів $D_i^{'}$ за допомогою НСУ δ_i, $i = \overline{1,2}$. Тобто $(L_1\theta)\delta_1$ та $(L_2\theta)\delta_2$ перетворюються в одну літеру. Якщо склейки не потрібно, вважаємо $\delta_i = \varepsilon$. Далі вводимо підстановку $\delta = \delta_1 \cup \delta_2$, яка буде НСУ для $L_1\theta$ і для $L_2\theta$, а також υ – НСУ для $L_1(\theta \circ \delta)$ та $\neg L_2(\theta \circ \delta)$. Підстановки δ та υ існують тому, що існує резольвента для $D_1^{'}$ та $D_2^{'}$, яка запишеться у вигляді $D^{'} = (A \vee B)(\theta \circ \delta \circ \upsilon)$.

Оскільки всі літери в $L_1\theta$ і $L_2\theta$ уніфікуються, то уніфікуються всі літери в L_1 та L_2, зокрема, підстановкою $\theta \circ \delta$. Нехай λ_1 – НСУ для $\{L_1^1, L_1^2, \cdots, L_1^r\}$ і λ_2 – НСУ для $\{L_2^1, L_2^2, \cdots, L_2^s\}$ і $\lambda = \lambda_1 \cup \lambda_2$. Тобто $L_1^1\lambda = L_1^2\lambda = \cdots L_1^r\lambda = L_1\lambda$ і $L_2^1\lambda = L_2^2\lambda = \cdots L_2^s\lambda = \neg L_2\lambda$. Візьмемо σ – НСУ для $L_1^1\lambda$ та $L_2^1\lambda$, тоді резольвенту D_1 та D_2 можна записати у вигляді $D = \big((A \vee B)\lambda\big)\sigma = (A \vee B)(\lambda \circ \sigma)$.

Із означення НСУ слідує, що знайдеться підстановка γ така, що $\theta \circ \delta = \lambda \circ \gamma$, тобто $\theta \circ \delta \circ \upsilon = \lambda \circ \gamma \circ \upsilon$. $D' = (A \vee B)(\theta \circ \delta \circ \upsilon) =$ $= (A \vee B)(\lambda \circ \gamma \circ \upsilon) = \big((A \vee B)\lambda\big)(\gamma \circ \upsilon)$. Тут $\gamma \circ \upsilon$ – деякий уніфікатор для $L_1\lambda$ та $\neg L_2\lambda$, а σ – їх НСУ, тому існує підстановка β така, що $\gamma \circ \upsilon = \sigma \circ \beta$. Отримаємо, $D' = \big((A \vee B)(\lambda \circ \sigma)\big)\beta = D\beta$. А це означає, що D' є прикладом D.

Лема доведена.

Теорема 5.5. (теорема про повноту методу резолюцій).

Множина диз'юнктів S суперечлива тоді і тільки тоді, коли існує резольвентне виведення порожнього диз'юнкта з S.

Доведення.

Достатність обґрунтовувалася раніше.

Необхідність. Нехай множина S суперечлива, тоді за теоремою 5.2 існує замкнуте скінченне семантичне дерево T, яке відповідає S.

Якщо T містить лише один кореневий вузол, то $\square \in S$, і твердження теореми виконується. Нехай T має більше одного вузла, тоді можна знайти такий вузол N, який породжує два спростовуючих вузла N_1 та N_2, адже в протилежному випадку кожен вузол мав би принаймні одного неспростовуючого сина. В цьому випадку можна побудувати нескінченну гілку, що суперечить скінченності замкнутого дерева T.

Нехай $I(N) = \{m_1, m_2, \cdots, m_n\}$, $I(N_1) = \{m_1, m_2, \cdots, m_n, m_{n+1}\}$, $I(N_2) = \{m_1, m_2, \cdots, m_n, \neg m_{n+1}\}$ – часткові H-інтерпретації, які відповідають вузлам N, N_1 та N_2. Оскільки N_1 та N_2 – спростовуючі вузли, а N – неспростовуючій, то існують два основні приклади D_1' та D_2' диз'юнктів D_1 та D_2, які хибні на $I(N_1)$ та $I(N_2)$ відповідно, але D_1' та D_2' не спростовуються на $I(N)$. Отже, D_1' містить $\neg m_{n+1}$, а D_2' – m_{n+1}. Відрізаючи літери $L_1' = \neg m_{n+1}$ та $L_2' = m_{n+1}$, можна отримати резольвенту D', яка буде хибною на $I(N)$, бо при видаленні L_1' з D_1' та L_2' з D_2' отримаємо хибні на $I(N)$ диз'юнкти. За лемою 5.5 існує така резольвента $D = R(D_1, D_2)$, що D' – основний приклад D. Нехай T_1 – замкнуте семантичне дерево для множини диз'юнктів $S_1 = S \cup \{D\}$, яке отримано з T видаленням всіх вузлів та ребер, які розташовані нижче першого вузла, в якому спростовується резольвента D'. Зрозуміло, що кількість вузлів в дереві T_1 менше, ніж в T. Продовжимо цей процес для T_1. Якщо в T_1 є більше одного вузла, то можна отримати нову резольвенту деяких двох диз'юнктів S_1. Додавши її до S_1, можемо отримати інше замкнуте семантичне дерево з меншою кількістю вузлів. В решті решт, за скінченну кількість кроків отримаємо замкнуте семантичне дерево, яке містить лише кореневий вузол. Це можливо, тільки коли вивели □. Отже, резольвентне виведення □ з суперечливої множини диз'юнктів S існує.

Кінець доведення.

5.5.5. Стратегія насичення рівня та стратегія викреслення

У попередньому пункті доведено повнота методу резолюцій. Але необмежене використання цього методу може викликати породження великої кількості зайвих, або надлишкових диз'юнктів. У той ж час, із зростанням потужності множини диз'юнктів S збільшуються шанси «загубити» один, або кілька диз'юнктів, без яких не отримаємо шукане резольвентне виведення.

Для збільшення ефективності використання методу резолюцій використовується стратегії насичення рівня та стратегія викреслення.

Стратегія насичення рівня. Нехай S – множина диз'юнктів. Покладемо $S_0 = S$ і для $n \in \mathbb{N}$ будуємо множину S_n за таким правилом.

$$S_n = \{R(C_1, C_2) | \, C_1 \in \bigcup_{i=0}^{n-1} S_i \, , C_2 \in S_{n-1}\}.$$

Множина S_n називається множиною диз'юнктів n-го рівня.

Приклад 5.39. Нехай $S = \{P \vee Q, P \vee \neg Q, \neg P \vee Q, \neg P \vee \neg Q\}$.

S_0	1.	$P \vee Q$	
	2.	$P \vee \neg Q$	
	3.	$\neg P \vee Q$	
	4.	$\neg P \vee \neg Q$	
		_____	
S_1	5.	P	$R(1,2)$
	6.	Q	$R(1,3)$
	7.	$Q \vee \neg Q$	$R(1,4)$
	8.	$P \vee \neg P$	$R(1,4)$
	9.	$Q \vee \neg Q$	$R(2,3)$

	10.	$P \vee \neg P$	$R(2,3)$
	11.	$\neg Q$	$R(2,4)$
	12.	$\neg P$	$R(3,4)$
S_2	13 – 16.	$P \vee Q$	$R(1,7) = R(1,8) = R(1,9) = R(1,10)$
	17.	P	$R(1,11)$
	18.	Q	$R(1,12)$
	19.	P	$R(2,6)$
	20 – 23.	$P \vee \neg Q$	$R(2,7) = R(2,8) = R(2,9) = R(2,10)$
	24.	$\neg Q$	$R(2,12)$
	25.	Q	$R(3,5)$
	26 – 29.	$\neg P \vee Q$	$R(3,7) = R(3,8) = R(3,9) = R(3,10)$
	30.	$\neg P$	$R(3,11)$
	31.	$\neg Q$	$R(4,5)$
	32.	$\neg P$	$R(4,6)$
	33 – 36.	$\neg P \vee \neg Q$	$R(4,7) = R(4,8) = R(4,9) = R(4,10)$
	37.	P	$R(5,8)$
	38.	P	$R(5,10)$
	39.	□	$R(5,12)$

Як бачимо, □ отримано лише на 39-му кроці. Крім того, диз'юнкти 7 – 10 – тавтології, або тавтологічні диз'юнкти, а також диз'юнкти 13 – 38 повторюють вище написані в S_0 та S_1.

Стратегія викреслення. Для того, щоб уникнути породження великої кількості надлишкових диз'юнктів ми можемо позбутися «дублікатів» вже наявних диз'юнктів, тавтологічних диз'юнктів, при

цьому суперечливість (несуперечливість) S зберігається, а також так званих наддиз'юнктів.

Означення 5.26. Нехай диз'юнкти D_1 та D_2 зв'язані співвідношенням $D_2 = D_1\sigma \vee C$, де C – деякий, можливо порожній, диз'юнкт, а σ – деяка підстановка. Тоді диз'юнкт D_2 називається наддиз'юнктом для D_1, або D_1 – піддиз'юнктом D_2.

Зрозуміло, що диз'юнкт D_1 буде спростовуватися на всіх інтерпретаціях, на яких спростовується D_2, тому при видаленні D_2 суперечливість (несуперечливість) множини диз'юнктів збережеться. Зрозуміло також, що кожен диз'юнкт є наддиз'юнктом (піддиз'юнктом) самого себе.

Вправа 5.5. Перевірити, чи є диз'юнкт D_2 наддиз'юнктом D_1:

$D_1 = P(x, y) \vee Q(z)$, $D_2 = Q(a) \vee P(b, b) \vee R(u)$;

$D_1 = P(x, y) \vee R(y, x)$, $D_2 = P(a, y) \vee R(z, b)$;

$D_1 = \neg P(x) \vee P\big(f(x)\big)$, $D_2 = \neg P(x) \vee P\left(f\big(f(x)\big)\right)$.

Сама стратегія викреслення полягає в тому, що видаляються всі тавтологічні диз'юнкти та наддиз'юнкти.

Зауваження 5.6. Якщо диз'юнкт є наддиз'юнктом лише самого себе, то він не викреслюється.

Стратегія викреслення буде повною, якщо її використовувати разом із стратегією насичення рівня наступним чином:

Спочатку виписуються диз'юнкти $S_0 \cup S_1 \cup \cdots \cup S_{n-1}$ по порядку; потім обчислюємо резольвенти шляхом порівняння кожного диз'юнкта $D_1 \in \bigcup_{i=0}^{n-1} S_i$ з диз'юнктом $D_2 \in S_{n-1}$, який записано після D_1. Саму резольвенту записуємо в кінець списку, якщо вона не тавтологія та не поглинається жодним диз'юнктом із списку. В протилежному випадку вона викреслюється.

Приклад 5.40. (Продовження прикладу 5.39).

Застосуємо ці стратегії для $S = \{P \vee Q, P \vee \neg Q, \neg P \vee Q, \neg P \vee \neg Q\}$ отримаємо список:

S_0	1.	$P \vee Q$	
	2.	$P \vee \neg Q$	
	3.	$\neg P \vee Q$	
	4.	$\neg P \vee \neg Q$	
		_____	
S_1	5.	P	$R(1,2)$
	6.	Q	$R(1,3)$
	7.	$\neg Q$	$R(2,4)$
	8.	$\neg P$	$R(3,4)$
		_____	
S_2	9.	$\square$	$R(5,8)$

Бачимо, що цей список значно коротший, ніж отриманий раніше.

Більш докладно про стратегії насичення рівня та викреслення, зокрема, про алгоритм перевірки, чи є один диз'юнкт наддиз'юнктом іншого, див. у [3].

Про мову програмування Пролог, яка ґрунтується на методі резолюцій можна прочитати, наприклад, в [7].

Розділ 6. Формальна арифметика

6.1. Побудова формальної теорії S

6.1.1. Система аксіом Пеано

Зауваження 6.1. В цьому розділі вважається, що множина натуральних чисел містить нуль.

Саме з арифметики натуральних чисел як найбільш безпосередньо інтуїтивної області математики почалися спроби формалізації та строгого обґрунтування математики. Першу, напіваксіоматичну побудову цієї дисципліни запропонував Дедекінд в 1901 році, яка стала відомою під назвою «Системи аксіом Пеано». Вона формулюється наступним чином:

P1: 0 є натуральним числом;

P2: для будь-якого натурального числа xіснує інше натуральне число, яке позначається x'та називається таким, що (безпосередньо) слідує за x;

P3: $0 \neq x'$ для будь-якого натурального числа;

P4: якщо $x' = y'$, то $x = y$;

P5: якщо Q – деяка властивість, якою можуть володіти одні, та не володіти інші натуральні числа, та якщо:

а) натуральне число 0 володіє властивістю Q та

б) для кожного натурального числа x з того, що x володіє властивістю Q, випливає, що й натуральне число x' володіє властивістю Q, то властивістю Q володіють усі натуральні числа (принцип індукції).

Цих аксіом, разом з деяким фрагментом теорії множин, достатньо для побудови не тільки арифметики, але і теорії раціональних, дійсних та

комплексних чисел. Але ця теорія не є строго формалізованою, оскільки містить інтуїтивні поняття, зокрема, «властивість». Тому на основі аксіоматики Пеано побудуємо деяку теорію першого порядку S, якої має вистачити для виведення всіх основних результатів елементарної арифметики.

6.1.2. Система аксіом формальної теорії S

Маємо злічену кількість змінних $x_1, x_2, \ldots$; єдину константу a_1, яку позначатимемо через 0; три функціональні символи: один унарний f_1^1 та два бінарних f_1^2 та f_2^2, які позначатимемо $f_1^1(t) = t'$; $f_1^2(t,s) = t + s$; $f_2^2(t,s) = t \cdot s$; а також один бінарний предикатний символ A_1^2, що позначається як $A_1^2(t,s)$: $t = s$ і $\neg A_1^2(t,s)$: $t \neq s$, де t та s – терми.

Крім п'яти логічних аксіом A1 – A5 та правил виведення MP та Gen (див. параграф 4.6) вводяться дев'ять власних аксіом:

S1: $(x_1 = x_2) \to \big((x_1 = x_3) \to (x_2 = x_3)\big)$;

S2: $(x_1 = x_2) \to (x_1' = x_2')$;

S3: $0 \neq (x_1)'$;

S4: $(x_1' = x_2') \to (x_1 = x_2)$;

S5: $x_1 + 0 = x_1$;

S6: $x_1 + x_2' = (x_1 + x_2)'$;

S7: $x_1 \cdot 0 = 0$;

S8: $x_1 \cdot x_2' = (x_1 \cdot x_2) + x_1$;

S9: $A(0) \to \big(\forall x\big(A(x) \to A(x')\big) \to \forall x\, A(x)\big)$,

де $A(x)$ – довільна формула теорії S.

Відмітимо, що аксіоми S1 – S8 є конкретними формулами, а S9 є системою аксіом, що породжує їх нескінченну кількість. Систему S9 називатимемо принципом математичної індукції, але вона не відповідає повністю аксіомі P5 тому, що в P5 інтуїтивно вважається $2^{\aleph_0}$ властивостей натуральних чисел, а S9 містить злічену кількість, які визначаються формулами теорії S.

Аксіоми S3 та S4 відповідають P3 та P4**,** наявність константи 0 та функціонального символу f_1^1 забезпечують P1 та P2 існування нуля та операції визначення наступного натурального числа. Аксіоми S1 та S2 Дедекіндом та Пеано вважалися інтуїтивно очевидними. Аксіоми S5 – S8 рекурсивно визначають операції додавання та множення. Жодних постулатів, які відповідають цим аксіомам в аксіоматиці Пеано немає. Дедекінд та Пеано допустили, що вони виводяться за допомогою P1 – P5 та інтуїтивної теорії множин.

Зауваження 6.2. Використовуючи A4, MP та Gen можна побудувати виведення $A(x) \vdash A(t)$, де t – довільний терм.

1.	$A(x)$	Г
2.	$\forall x\, A(x)$	Gen (1)
3.	$\forall x\, A(x) \to A(t)$	A4
4.	$A(t)$	MP (2, 3)

Аксіоми S1 – S8 можемо переписати в такому вигляді:

S1′: $(t = r) \to \big((t = s) \to (r = s)\big)$;

S2′: $(t = r) \to (t' = r')$;

S3′: $0 \neq t'$;

S4′: $(t' = r') \to (t = r)$;

S5′: $t + 0 = t$;

S6′: $t + r' = (t + r)'$;

S7′: $t \cdot 0 = 0$;

S8′: $t \cdot r' = (t \cdot r) + t$.

6.1.3. Приклади доведення в теорії S

Наведемо декілька прикладів доведення в теорії S деяких тверджень елементарної арифметики.

Теорема 1. $\vdash t = t$ (рефлексивність).

1.	$t + 0 = t$	S5′
2.	$(t + 0 = t) \to \big((t + 0 = t) = (t = t)\big)$	S1′
3.	$(t + 0 = t) \to (t = t)$	MP (1,2)
4.	$t = t$	MP (1, 3)

Теорема 2. $\vdash (t = r) \to (r = t)$ (симетричність).

1.	$(t = r) \to \big((t = t) \to (r = t)\big)$	S1′
2.	$t = t$	T1
3.	$(t = r) \to (r = t)$	B2 (1, 2)

Теорема 3. $\vdash (t = r) \to \big((r = s) \to (t = s)\big)$ (транзитивність).

1.	$(r = t) \to \big((r = s) \to (t = s)\big)$	S1′
2.	$(t = r) \to (r = t)$	T2
3.	$(t = r) \to \big((r = s) \to (t = s)\big)$	B1 (2, 1)

Теорема 4. $\vdash (r = t) \to \big((s = t) \to (r = s)\big)$.

1.	$(r = t) \to \big((t = s) \to (r = s)\big)$	T3
2.	$(t = s) \to \big((r = t) \to (r = s)\big)$	B4 (1)

3. $(s = t) \to (t = s)$ T2

4. $(s = t) \to \big((r = t) \to (r = s)\big)$ B1(3, 2)

5. $(r = t) \to \big((s = t) \to (r = s)\big)$ B4 (4)

Правила виведення B1, B2 та B4 взято з теорії L (див. параграф 3.2).

Зауваження 6.3. Для скорочення запису домовимося замість виведення

a. A_1

b. $A_1 \to A_2$ T

c. A_2 MP (a, b)

будемо писати виведення

a. A_1

b. A_2 T (a, b),

та замість виведення

a. A_1

b. A_2

c. $A_1 \to (A_2 \to A_3)$ T

d. $A_2 \to A_3$ MP (a, c)

e. A_3 MP (b, d)

писатемемо

a. A_1

b. A_2

c. A_3 T (a, b).

Теорема 5. $\vdash (t = r) \to (t + s = r + s)$.

Застосуємо правило індукції до формули $A(z)$: $(x = y) \to (x + z = y + z)$.

I. Спочатку доведемо $\vdash A(0)$.

$$\vdash A(0) \Leftrightarrow \vdash (x = y) \to (x + 0 = y + 0) \overset{\text{МТД}}{\Leftrightarrow} x = y \vdash x + 0 = y + 0.$$

1.	$x = y$	Г1
2.	$x + 0 = x$	S5′
3.	$y + 0 = y$	S5′
4.	$y = y + 0$	Т2 (3) (див. зауваження 6.3)
5.	$x + 0 = y$	Т3 (2, 1)
6.	$x + 0 = y + 0$	Т3 (5, 4).

II. Далі побудуємо виведення $\vdash A(z) \to A(z')$.

$$\vdash A(z) \to A(z') \Leftrightarrow$$

$$\vdash \big((x = y) \to (x + z = y + z)\big) \to \big((x = y) \to (x + z' = y + z')\big) \overset{\text{МТД}}{\Leftrightarrow}$$

$$\overset{\text{МТД}}{\Leftrightarrow} (x = y) \to (x + z = y + z) \vdash (x = y) \to (x + z' = y + z') \overset{\text{МТД}}{\Leftrightarrow}$$

$$\overset{\text{МТД}}{\Leftrightarrow} (x = y) \to (x + z = y + z), x = y \vdash x + z' = y + z'$$

1.	$(x = y) \to (x + z = y + z)$	Г1
2.	$x = y$	Г2
3.	$x + z = y + z$	MP (2, 1)
4.	$(x + z)' = (y + z)'$	S2′ (3)
5.	$x + z' = (x + z)'$	S6′
6.	$x + z' = (y + z)'$	Т3 (5, 4)
7.	$y + z' = (y + z)'$	S6′
8.	$x + z' = y + z'$	S6 (6, 7)

Отримали $\vdash A(0)$ та $\vdash A(z) \to A(z')$. За допомогою правила Gen матимемо $\vdash \forall z \big(A(z) \to A(z')\big)$. Використовуємо S9 і отримаємо $\vdash \forall z\, A(z)$. Застосуємо правила Gen, MP та $A4$ отримаємо твердження даної теореми (див. зауваження 6.2). $\vdash (t = r) \to (t + s = r + s)$.

Вправа 6.1. Довести теореми в теорії S:

T6 $\vdash t = 0 + t$;

T7 $\vdash t' + r = (t + r)'$;

T8 $\vdash t + r = r + t$;

T9 $\vdash (t = r) \to (s + t = s + r)$;

T10 $(t + r) + s = t + (r + s)$;

T11 $\vdash (t = r) \to (t \cdot s = r \cdot s)$;

T12 $\vdash 0 \cdot t = 0$;

T13 $\vdash t' \cdot r = t \cdot r + r$;

T14 $\vdash t \cdot r = r \cdot t$;

T15 $\vdash (t = r) \to (s \cdot t = s \cdot r)$.

Інші приклади доведення в формальній теорії S див, наприклад, в [1].

Означення 6.1. Інтерпретація теорії S, в якій

1. Предмета область – $\mathbb{N} \cup \{0\} = \mathbb{N}_0$;
2. Ціле число 0 інтерпретує символ 0;
3. Операція взяття наступного числа (додавання одиниці) інтерпретує функціональний символ $f_1^1 = '$;
4. Звичайні операції додавання та множення інтерпретують $+$ та $\cdot$;
5. Предикатний символ = інтерпретується тотожним відношенням.

називається стандартною моделлю теорії S. Всі інші моделі не ізоморфні стандартній, називаються нестандартними моделями.

6.2. Арифметичні функції та відношення теорії S

У цьому параграфі будемо розглядати відношення $R \in \mathbb{N}_0^{n\,\mathrm{n}}$ та функції $f\colon \mathbb{N}_0^n \to \mathbb{N}_0$, $n \in \mathbb{N}$.

Введено позначення: $\bar{1} = 0'$, $\bar{2} = 0''$, … $\overline{k+1} = (\bar{k})'$, … .

Означення 6.2. Характеристичною функцією відношення $R \in \mathbb{N}_0^n$ називається функція $\chi_R\colon \mathbb{N}_0^n \to \{0;1\}$ така, що

$$\chi_R(x_1, x_2, \dots, x_n) = \begin{cases} 1, & (x_1, x_2, \dots, x_n) \in R \\ 0, & (x_1, x_2, \dots, x_n) \notin R \end{cases}.$$

Далі замість $(x_1, x_2, \dots, x_n) \in R$ будемо писати $R(x_1, x_2, \dots, x_n)$, та замість $(x_1, x_2, \dots, x_n) \notin R$ будемо писати $\neg R(x_1, x_2, \dots, x_n)$. Також будемо ототожнювати поняття відношення та відповідного предикату $R\colon \mathbb{N}_0^n \to \{0, 1\}$.

Означення 6.3. Арифметичне відношення $R \in \mathbb{N}_0^n$ зображується в теорії S, якщо існує формула $\mathcal{A}(x_1, x_2, \dots, x_n)$ теорії S з n вільними змінними така, що для всіх $k_1, k_2, \dots, k_n \in \mathbb{N}_0$ виконуються дві умови:

1. Якщо $R(k_1, k_2, \dots, k_n)$, то $\vdash_S \mathcal{A}(\bar{k}_1, \bar{k}_2, \dots, \bar{k}_n)$;
2. Якщо $\neg R(k_1, k_2, \dots, k_n)$, то $\vdash_S \neg\mathcal{A}(\bar{k}_1, \bar{k}_2, \dots, \bar{k}_n)$.

Приклад 6.1. Відношення рівності між натуральними числами зображується в S формулою $x_1 = x_2$. Дійсно, якщо $k_1 = k_2$, то $\bar{k}_1 = \bar{k}_2$ і за теоремою 1 $\vdash_S \bar{k}_1 = \bar{k}_2$. Якщо ж $\bar{k}_1 \neq \bar{k}_2$, то в наслідок S3′, S4′ та теореми 2 $\vdash_S \neg(\bar{k}_1 = \bar{k}_2)$.

Вправа 6.2. Довести, що відношення " $\geq$ " зображується в S за допомогою формули $\mathrm{A}(x_1, x_2)\colon \exists y(x_1 = x_2 + y)$.

Запис $\exists_1 x A(x) = \exists x A(x) \wedge \forall x_1 \forall x_2 \Big(\big(A(x_1) \wedge A(x_2) \big) \to (x_1 = x_2) \Big)$ означатиме висловлення «існує єдиний x такий, що формула $A(x)$ правдива».

Означення 6.4. Арифметична функція $f(x_1, \dots, x_n)$ зображується в теорії S, якщо існує формула $\mathcal{A}(x_1, x_2, \dots, x_{n+1})$ теорії S з вільними змінними $x_1, x_2, \dots, x_{n+1}$ така, що для всіх $k_1, k_2, \dots, k_{n+1} \in \mathbb{N}_0$ виконуються дві умови:

1. Якщо $f(k_1, k_2, \dots, k_n) = k_{n+1}$, то $\vdash_S \mathcal{A}(\bar{k}_1, \bar{k}_2, \dots, \bar{k}_{n+1})$
2. $\vdash_S \exists_1 x_{n+1} A(x_1, x_2, \dots, x_n, x_{n+1})$.

Якщо ж умову 2 замінити умовою

$2'$. $\vdash_S \exists_1 x_{n+1} A(x_1, x_2, \dots, x_n, x_{n+1})$,

то отримаємо означення функції, яка сильно зображується в теорії S.

Відмітимо, що з правила Gen та A4 з $2'$ слідує 2. Отже, будь-яка функція, що сильно зображується в S буде зображуватися в S.

Приклад 6.2. Анулятор $O(x) = 0$ сильно зображується в S за допомогою формули $(x_1 = x_2) \wedge (x_2 = 0)$. Дійсно, умова 1 виконується. Якщо $O(k_1) = k_2$ і $k_2 = 0$ $\vdash_S \big(\bar{k}_1 = \bar{k}_1\big) \wedge (0 = 0)$. Також очевидно, що $\vdash_S \exists_1 x_1 \big((x_1 = x_2) \wedge (x_2 = 0) \big)$ – умова $2'$.

Приклад 6.3. Інкремент $P(x) = x + 1$ також сильно зображується в S формулою $x_2 = x_1'$. Дійсно, якщо $k_2 = k_1 + 1$, то $\vdash k_2 = (k_1)'$. Також $\vdash_S \exists_1 x_2 (x_2 = x')$.

Приклад 6.4. Проекція $I_m^n(x_1, x_2, \dots, x_n) = x_m$ сильно зображується в S за допомогою формули

$(x_1 = x_1) \wedge (x_2 = x_2) \wedge \dots \wedge (x_n = x_n) \wedge (x_{n+1} = x_m)$.

Приклад 6.5. Припустимо, що функції $g^{(m)}, h_1^{(n)}, h_2^{(n)}, \dots, h_m^{(n)}$ (сильно) зображуються в S формулами $\mathcal{B}(x_1, x_2, \dots, x_{m+1})$,

$\mathcal{A}_1(x_1, x_2, \dots x_{n+1})$, …, $\mathcal{A}_m(x_1, x_2, \dots x_{n+1})$ відповідно. Їх суперпозиція $f(x_1, x_2, \dots, x_n) = g\big(h_1(x_1, x_2, \dots, x_n), \dots, h_m(x_1, x_2, \dots, x_n)\big)$ (сильно) зображується в S, наприклад, за допомогою формули

$$\exists y_1 \exists y_2 \dots \exists y_n (\mathcal{A}_1(x_1, \dots, x_n, y_1) \wedge \dots \wedge \mathcal{A}_m(x_1, \dots, x_n, y_m) \wedge \\ \wedge \mathcal{B}(y_1, \dots, y_m, x_{n+1}).$$

Теорема 6.1. Якщо відношення $R \subset \mathbb{N}^n$ зображується в S, то його характеристична функція $\mathcal{X}_R$ сильно зображується в S, а якщо функція $\mathcal{X}_R$ зображується в S, то в S зображується і саме відношення R.

Доведення.. Нескладно перевірити, що якщо R зображується в S за допомогою формули $\mathcal{A}(x_1, x_2, \dots, x_n)$, то функція $\mathcal{X}_R(x_1, x_2, \dots, x_n)$ сильно зображується в S за допомогою формули

$$\big(\mathcal{A}(x_1, x_2, \dots, x_n) \wedge (x_{n+1} = 1)\big) \vee \big(\neg\mathcal{A}(x_1, x_2, \dots, x_n) \wedge (x_{n+1} = 0)\big),$$

а також, якщо функція $\mathcal{X}_R(x_1, x_2, \dots, x_n)$ зображується в S за допомогою формули $\mathcal{B}(x_1, x_2, \dots, x_{n+1})$, то відношення R зображується в S за допомогою формули $\mathcal{B}(x_1, x_2, \dots, x_n, 1)$.

Кінець доведення.

Означення 6.5. Відношення називають примітивно рекурсивним (частково рекурсивним, загально рекурсивним), якщо його характеристична функція є ПРФ (ЧРФ, ЗРФ).

Теорема 6.2. Будь яка примітивно рекурсивна функція сильно зображується в S. Будь-яка загально рекурсивна функція зображується в S.

Доведення. Твердження теореми випливає з того, що вихідні функції сильно зображуються в S (приклади 6.2 – 6.4) та операція взяття суперпозиції (приклад 6.3) не виводить за рамки сильної зображуваності. Те, що схема примітивної рекурсії не виводить за рамки сильної

зображуваності та всюди визначений μ–оператор не виводить за рамки зображуваності див. [1, с. 146-150].

Кінець доведення.

Наслідок. Будь-яке примітивно рекурсивне відношення сильно зображується в S. Будь-яке загально рекурсивне відношення зображується в S.

Доведення твердження випливає з теореми 6.1 та теореми 6.2.

6.3. Арифметизація логіки. Геделеві номери

Гедель запропонував спосіб кодування символів, виразів та послідовностей виразів так, щоб кожному такому об'єкту однозначно відповідало деяке натуральне число. Такий спосіб кодування отримає назву «геделева нумерація».

Кожному символу u довільної теорії першого порядку K поставимо у відповідність натуральне число $g(u)$, яке називається геделевим номером символа u:

$g(() = 3;$

$g()) = 5;$

$g(,) = 7;$

$g(\neg) = 9;$

$g(\rightarrow) = 11;$

$g(x_k) = 5 + 8k;$

$g(a_k) = 7 + 8k;$

$g(f_n^k) = 9 + 8 \cdot 2^n \cdot 3^k;$

$g(A_k^n) = 11 + 8 \cdot 2^n \cdot 3^k;$

Тут всюди $n, k \in \mathbb{N}$, x_k – k-та змінна, a_k – k–та константа, f_k^n – k–тий n–арний функціональний символ, A_k^n – k–тий n–арний предикатний символ. Також замість $\forall x_i$ будемо писати (x_i).

Якщо вираз w складається із символів $\alpha_1, \alpha_2, \dots, \alpha_n$ $(w = \alpha_1\alpha_2 \dots \alpha_n)$, то $g(w) = p_0^{g(\alpha_1)} p_1^{g(\alpha_2)} \dots p_{n-1}^{g(\alpha_n)}$, де p_n – n–те просте число $(p_0 = 2,\ p_1 = 3,\ p_2 = 5, \dots)$.

Для послідовності виразів $w_1, w_2, \dots, w_m$, $m \in \mathbb{N}$

$$g(w_1, w_2, \dots, w_m) = p_0^{g(w_1)} p_1^{g(w_2)} \dots p_{m-1}^{g(w_m)}$$

Бачимо, що різні символи, різні вирази та різні послідовності виразів мають різні геделеві номери. Кожен символ має непарний номер, а кожен вираз – парний (навіть у випадку, коли вираз складається з одного символу). Для виразів геделевий номер має двійку в непарному степені, а для послідовності виразів – двійку в парному степені (навіть, коли в послідовності є лише один вираз).

Приклад 6.6. $g(x_3) = 29$, $g(a_1) = 15$, $g(f_2^1) = 153$, $g(A_1^3) = 203$,

$$g(A_1^2(x_1, x_2) = 2^{g(A_1^2)} \cdot 3^{g(} \cdot 5^{g(x_1)} \cdot 7^{g(,)} \cdot 11^{g(x_2)} \cdot 13^{g())} =$$
$$= 2^{107} \cdot 3^3 \cdot 5^{13} \cdot 7^7 \cdot 11^{21} \cdot 13^5.$$

Але множина значеннь функції g не співпадає з множиною натуральних чисел. Зокрема, число 12 не є геделевим номером.

Вправа 6.3. Побудувати об’єкти, які мають геделеві номери 1944 та 63.

Вправа 6.4. Знайти геделеві номери виразів $A_1^2(x_1, a_1)$ та $A_1^2(x_1, a_1) \to A_1^1(x_3)$.

Така нумерація вперше використана Геделем в 1931 році з метою арифметизації логіки, тобто з метою заміни тверджень про формальну систему еквівалентними висловленнями про натуральні числа з подальшим перенесенням цих висловлень до формальних систем. Ідея арифметизації стала ключем до розв'язування багатьох важливих проблем математичної логіки.

У [1] доведена примітивна рекурсивність цілої низки предикатів. Серед них є наступні.

Лема 6.1. В теорії S такі предикати є примітивно рекурсивними:

1. Нехай $\mathcal{A}(x_1, x_2, \dots, x_n)$ – деяка фіксована формула теорії S, яка має єдині вільні змінні $x_1, x_2, \dots, x_n$ і нехай $m = g\big(\mathcal{A}(x_1, x_2, \dots, x_n)\big)$. Позначимо через $B_{W_A}(u_1, u_2, \dots, u_n, y)$ висловлення: «y є геделевим номером виведення в S формули $\mathcal{A}(\bar{u}_1, \bar{u}_2, \dots, \bar{u}_n)$».
2. а) $W_1(u, y)$: «k є геделевим номером формули $\mathcal{A}(x_1)$, що містить вільну змінну x_1 та y – геделевий номер виведення в S формули $\mathcal{A}(\bar{u})$»;

   б) $W_2(u, y)$: «k є геделевим номером формули $\mathcal{A}(x_1)$, що містить вільну змінну x_1 та y – геделевий номер виведення в S формули $\neg\mathcal{A}(\bar{u})$».

Теорема 6.3. Будь-яка функція $f(x_1, x_2, \dots, x_n)$, що зображується в S, є загально рекурсивною.

Доведення. Нехай формула $\mathcal{A}(x_1, x_2, \dots, x_n, z)$ зображує f в S. Розглянемо натуральні числа $k_1, k_2, \dots, k_n$. Нехай $f(k_1, k_2, \dots, k_n) = m$. Тоді $\vdash_s \mathcal{A}\big(\bar{k}_1, \bar{k}_2, \dots, \bar{k}_n, \bar{m}\big)$, і нехай j – геделевий номер виведення в S формули $\mathcal{A}\big(\bar{k}_1, \bar{k}_2, \dots, \bar{k}_n, \bar{m}\big)$. Тоді виконується $B_{W_A}(k_1, k_2, \dots, k_n, m, j)$ (див. лему 6.1, пункт 1) отже для всіх $x_1, x_2, \dots, x_n$ існує такий y, що $B_{W_A}\big(k_1, k_2, \dots, k_n, \exp(0, y), \exp(1, y)\big)$ правдивий. Оскільки $y = 2^m 3^j$,

функція $\exp(x, y)$ – показник степеня біля x–го простого числа в розкладі числа y на прості множники. Щоб знайти y використаємо μ–оператор. Тому $f(x_1, x_2, \dots, x_n) = \mu_y\left[B_{W_A}\left(k_1, k_2, \dots, k_n, \exp(0, y), \exp(1, y)\right) = 1\right]$. Внаслідок леми 6.1 предикат B_{W_A} є примітивно рекурсивним, а функція f є ЗРФ, оскільки тут використали всюди визначений μ-оператор.

Кінець доведення

З теорем 6.2 та 6.3 випливає теорема 6.4.

Теорема 6.4. Клас загально рекурсивних функцій співпадає з класом функцій, які зображуються в S.

Наслідок. Будь-яке задане на множині натуральних чисел відношення є загально рекурсивним тоді, і тільки тоді, коли воно зображується в теорії S.

6.4. Теорема Геделя про неповноту теорії S

Розглянемо теорію першого порядку K з такими ж самими символами, як і в S.

Означення 6.6. Теорія K називається ω-несуперечливою, якщо для будь-якої формули $\mathcal{A}(x)$ цієї теорії з того, що для довільного $n \in \mathbb{N}_0$ $\vdash_K \mathcal{A}(\bar{n})$, випливає $\nvdash_K \exists x \neg \mathcal{A}(x)$ (формулу $\exists x \neg \mathcal{A}(x)$ в теорії K вивести неможна).

Лема 6.2. Якщо теорія K ω-несуперечлива, то вона несуперечлива.

Доведення. Нехай K одночасно є ω-несуперечливою та суперечливою. Згадаємо теорему 5 формальної теорії L: $\vdash \neg A \to (A \to B)$. Тобто, якщо формула виводиться разом із своїм запереченням, то будь-яка

формула є теоремою. Розглянемо довільну формулу, яка виводиться в K, наприклад, $(x = x) \to (x = x)$. Для будь-якого $n \in \mathbf{N}$ очевидно, що $\vdash_K (\bar{n} = \bar{n}) \to (\bar{n} = \bar{n})$, тому формула $\exists x \neg \big((x = x) \to (x = x)\big)$ не виводиться в K, що суперечить тому, що в K має виводитися будь-яка формула.

Лема доведена.

Згідно з лемою 6.1 (2 а)) відношення $W_1(u, y)$ є примітивно рекурсивним і зображується в S деякою формулою $\mathcal{W}_1(x_1, x_2)$ з двома вільними змінними x_1 та x_2. Це означає що, якщо $W_1(k_1, k_2)$ правдиве, то $\vdash_S \mathcal{W}_1\big(\bar{k}_1, \bar{k}_2\big)$, та , якщо $W_1(k_1, k_2)$ хибне, то $\vdash_S \neg\mathcal{W}_1\big(\bar{k}_1, \bar{k}_2\big)$. Розглянемо формулу

$$\forall x_2 \neg \mathcal{W}_1(x_1, x_2). \qquad (*)$$

І нехай m – геделевий номер формули $(*)$. Підставивши в $(*)$ $\bar{m}$ замість x_1, отримаємо замкнуту формулу

$$\forall x_2 \neg \mathcal{W}_1(\bar{m}, x_2). \qquad (**)$$

Згадаємо, що твердження $W_1(u, y)$ правдиве тоді і тільки тоді, коли $u = g\big(\mathcal{A}(x_1)\big)$, де x_1 – вільна змінна формули $\mathcal{A}(x_1)$, а y – геделевий номер виведення в S формули $\mathcal{A}(\bar{u})$.

Отже, (і) $W_1(m, y)$ правдиве тоді, і тільки тоді, коли y – геделевий номер виведення в S формули $(**)$.

Теорема 6.5. (теорема Геделя для теорії S).

1. Якщо теорія S несуперечлива, то формула $(**)$ не виводиться в S.
2. Якщо теорія S ω-несуперечлива, то формула $\neg(**)$ не виводиться в S.

Доведення.

1. Припустимо, що теорія S несуперечлива і $\vdash_S \forall x_2 \neg \mathcal{W}_1(\overline{m}, x_2)$. Нехай k – геделевий номер якого-небудь виведення в S останньої формули. Внаслідок (i), справджується $W_1(m, y)$. Оскільки $\mathcal{W}_1$ зображує в S формулу W_1 в S, то $\vdash_S \mathcal{W}_1(\overline{m}, \overline{k})$. Але з $\forall x_2 \neg \mathcal{W}_1(\overline{m}, x_2)$ за системою аксіом A4 можна вивести $\neg \mathcal{W}_1(\overline{m}, \overline{k})$, що суперечить припущенню про несуперечливість теорії S.

2. Нехай теорія S ω- несуперечлива та $\vdash_S \neg \forall x_2 \neg \mathcal{W}_1(\overline{m}, x_2)$, тобто $\vdash_S \neg(**)$. З леми 6.2 слідує, що S несуперечлива і $\nvdash_S (**)$. Отже $\forall n \in \mathbb{N}_0$, n не є геделевим номером виведення в S формули $(**)$, тобто $W_1(m, n)$ хибне для всіх n. Взявши в якості $\mathcal{A}(x_2)$ формулу $\neg \mathcal{W}_1(\overline{m}, x_2)$ та врахувавши припущення про ω- несуперечливість теорії S, приходимо до висновку, що $\nvdash_S \exists x_2 \neg\neg \mathcal{W}_1(\overline{m}, x_2)$, а тому $\nvdash_S \exists x_2 \mathcal{W}_1(\overline{m}, x_2)$. Ми отримали суперечність з умовою $\vdash_S \exists x_2 \mathcal{W}_1(\overline{m}, x_2)$.

Кінець доведення.

Розглянемо стандартну інтерпретацію формули $(**)$, яка не доводиться та не спростовується. Оскільки $\mathcal{W}_1$ зображує в S відношення W_1, то, у відповідності із стандартною інтерпретацією, $(**)$ стверджує, що $W_1(m, x_2)$ хибне для кожного натурального числа x_2. Внаслідок (i), не існує виведення формули $(**)$ в S. Іншими словами, $(**)$ стверджує те, що вона не виводиться в S. За теоремою Геделя, якщо теорія S несуперечлива, ця формула насправді не виводиться в S і тому правдива на стандартній інтерпретації. Тобто формула $(**)$ правдива, але в S не виводиться. Це може наштовхнути на думку, теорема Геделя справедлива для теорії S тому, що початково вибрана для даної теорії система аксіом виявилася занадто слабкою, і додання до S нових аксіом може призвести до того, що

нова теорія стане повною. Наприклад, можна утворити сильнішу теорію S_1, додавши до S правдиву формулу $(**)$. Але довільна загально рекурсивна функція, яка зображується в S, зображатиметься і в теорії S_1. А тому лему 6.1 можна переформулювати для S_1. Цього достатньо, щоб отримати результат Геделя для S_1. Тобто з ω- несуперечливості S_1 слідує наявність у ній формули $\mathcal{B}$, яка не доводиться те не спростовується. Вигляд $\mathcal{B}$ аналогічний формулі $(**)$ ($\mathcal{B} = \forall x_2 \neg (\mathcal{W}_1)_{S_1}(\bar{k}, x_2)$). Але $\mathcal{B}$ відрізняється від $(**)$, оскільки відношення W_1 для S_1 не співпадає з відношення W_1 для S, і тому, формула $(\mathcal{W}_1)_{S_1}$ та цифра $\bar{k}$, які містяться в $\mathcal{B}$, відрізнятимуться від формули $\mathcal{W}_1$ та цифри $\bar{m}$, які містяться в $(**)$.

Россер показав, що можна обмежитися припущенням про про звичайну несуперечливість теорії S, ціною деякого ускладнення доведення.

Гедель також довів, що, якщо теорія S несуперечлива, то доведення її несуперечливості неможна провести засобами самої теорії S, тобто для цього потрібні ідеї, або методи, що не відтворюються в S. Прикладами таких доведень можуть слугувати доведення несуперечливості S запропоновані Генценом в 1936 і 1938 роках та Шютте в 1951 році, яке можна знайти в [1]. В цих доведеннях застосовуються поняття та методи, які не формалізуються засобами теорії S, зокрема, фрагменти теорії злічених порядкових чисел.

Список літератури

1. Мендельсон Э. Введение в математическую логику / Мендельсон Э. — М. : Наука, 1984. — 320 с.
2. Клини С.К. Математическая логика / Клини С.К. — М. : Мир, 1973. — 480 с.
3. Чень Ч. Математическая логика и автоматическое доказательство теорем. / Чень Ч., Ли Р. — М. : Наука, 1983. — 360 с.
4. Таран Т.А. Основы дискретной математики. / Таран Т.А. — К. : Просвіта, 2003. — 288 с.
5. Лихтарников Л.М. Математическая логика. Курс лекций. Задачник-практикум и решения. / Лихтарников Л.М., Сукачева Т.Г. — СПб. : Лань, 1999. — 288 с.
6. Спекторський І.Я. Дискретна математика / І.Я. Спекторський. — К.: ІВЦ «Політехніка», 2004. — 220 с.
7. Иван Братко. Алгоритмы искусственного интеллекта на языке PROLOG. Третье издание.: Пер. с анг. / Братко И. — М.: Издательский дом «Вильямс», 2004. — 640 с.

Покажчик термінів